# *HYPERSIGNAL-MACRO TUTORIAL*

***(revision 4.0)***
***March, 2000***

***Signalogic, Inc.***
***9617 Wendell @ Skillman***
***Dallas, TX 75243***
***tel: 214-343-0069***
***fax: 214-343-0163***

***http//: www.signalogic.com***
***tech support: dsphelp@signalogic.com***

# TABLE OF CONTENTS

1.0 INTRODUCTION . . . . . 1-1
2.0 GETTING STARTED . . . . . 2-1
2.1.1 Installing Hypersignal from CD . . . . . 2-1
2.1.2 Installing Hypersignal from floppy Disk . . . . . 2-1
2.2 Running and Initializing Hypersignal . . . . . 2-2
2.3 Configuring Hypersignal . . . . . 2-5
3.0 USING HYPERSIGNAL . . . . . 3-1
3.1 On-Line Help . . . . . 3-1
3.2 Data-Entry Menus . . . . . 3-2
3.2.1 Waveform Summaries . . . . . 3-4
3.2.2 Pop-Up Waveform Directories . . . . . 3-4
3.3 Using the Printer . . . . . 3-5
3.4 Saving the User Environment . . . . . 3-6
3.5 Invoking a DOS Shell . . . . . 3-8
3.6 Invoking External Programs . . . . . 3-8
4.0 ACQUIRING DATA . . . . . 4-1
4.1 Acquiring Data From Disk Files . . . . . 4-1
4.2 Acquiring Data From DSP/Analog Boards . . . . . 4-4
5.0 DISPLAYING / EDITING WAVEFORMS . . . . . 5-1
5.1 Display, Zoom, Pan, and Scaling . . . . . 5-3
5.1.1 Dual- and Overlaid-Trace Display . . . . . 5-7
5.1.2 Time Domain Log Amplitude Display . . . . . 5-10
5.2 Time Domain Waveform Editing . . . . . 5-11
5.3 Hardcopy Print-Out . . . . . 5-13
5.4 D/A Playback From Displays . . . . . 5-16
6.0 FFT GENERATION . . . . . 6-1
7.0 DISPLAYING FREQUENCY DOMAIN WAVEFORMS . . 7-1

TABLE OF CONTENTS


7.1 2-D Spectrograph (Contour) Display . . . . . 7-1
7.2 3-D Spectrogram (Waterfall) Display . . . . . 7-5
7.3 Magnitude Display . . . . . 7-9
7.4 Phase Unwrapping . . . . . 7-14
8.0 DIFFERENCE EQUATIONS . . . . . 8-1
8.1 Waveform Generation . . . . . 8-1
8.2 Signal Arithmetic . . . . . 8-10
8.3 Sample Manipulation . . . . . 8-18
9.0 DESIGNING FILTERS . . . . . 9-1
9.1 Classical IIR Filters . . . . . 9-2
9.2 Arbitrary IIR FIlters . . . . . 9-11
9.3 Classical FIR Filters . . . . . 9-15
9.4 Arbitrary FIR Filters . . . . . 9-18
10.0 APPLYING FILTERS . . . . . 10-1
10.1 Recursive Filtering . . . . . 10-1
10.2 Convolution . . . . . 10-5
10.3 “Snap-In” Real-Time Filtering . . . . . 10-10
11.0 SPECTRUM ANALYZER . . . . . 11-1
11.1 Sources and Combinations of Input . . . . . 11-1
11.1.1 Real-Time Filtering of Analog Input . . . . . 11-3
11.1.2 Dual- and Overlaid-Trace Display . . . . . 11-5
11.2 Time Measurement . . . . . 11-7
11.3 Frequency Measurement . . . . . 11-11
11.3.1 Log and Octave Band Frequency Axis Display . 11-23
11.4 Transfer Function Measurement . . . . . 11-28
11.4.1 Impulse Response Measurement . . . . . 11-36
11.5 Sampling Rate and Input Range . . . . . 11-40
11.6 Timebase, Triggering, Vertical Scaling & Position, and Other Controls . . . . . 11-42

11.7 Spectrum Analyzer Calibration . . . . . 11-45
11.8 3-D Map Mode Display . . . . . 11-50
12.0 STIMULUS & RESPONSE . . . . . 12-1
12.1 Generating Stimuli . . . . . 12-1
12.2 Simultaneous Input/Output . . . . . 12-6
12.3 Time Domain Analysis . . . . . 12-9
12.4 Frequency Domain Analysis . . . . . 12-16
12.5 Extended Time Domain Analysis . . . . . 12-18
13.0 TRANSFER FUNCTION ANALYSIS . . . . . 13-1
14.0 HILBERT TRANSFORMS . . . . . 14-1
14.1 Hilbert Transform Design Example . . . . . 14-1
14.2 Waveform Display/Edit Verification . . . . . 14-4
14.3 Digital Oscilloscope Verification . . . . . 14-9
15.0 MACRO EXAMPLES . . . . . 15-1
15.0.1 Macro Source, Object, and Configuration Files . 15-2
15.1 Sampling Rate Conversion . . . . . 15-4
15.2 Frequency Zoom . . . . . 15-10
15.3 Analog Conversion Macros . . . . . 15-18
15.4 Stimulus & Response Macro . . . . . 15-20
15.5 Interactive Display Macros . . . . . 15-24
15.5.1 Waveform Display/Edit Cut & Paste Macro . . . 15-24
15.5.2 Waveform Display/Edit Write to File Macro . . 15-28
15.5.3 2-D Spectrograph D/A Output Macro . . . . . 15-32
15.5.4 Instrumentation Display Macros . . . . . 15-36
15.6 Invoking External Programs From a Macro. . . . . 15-38
16.0 LPC AUTOCORRELATION . . . . . 16-1
17.0 FILTER QUANTIZATION EFFECTS . . . . . 17-1

# 1.0 INTRODUCTION

Welcome to the Hypersignal-Macro software tutorial. The tutorial also covers Hypersignal-Acoustic and Hypersignal-Macro NRT (non-real-time), as well as the Nihon versions (NEC PC980x versions) of each of these packages.

The tutorial is designed to get you up and running with Hypersignal software as quickly as possible. For a quick reference to Hypersignal functions, which are organized in general categories as Time Domain, Frequency Domain, and Utilities, please see the Software Reference Guide. For technically detailed and thorough discussion of the features and operation of Hypersignal, please refer to the Users Manual. For detailed technical information on specific DSP/Acquisition hardware, and also the various “DSP Source Code Interfaces”, see the Hardware Reference Guide.

Updates or changes to all documentation and recent information concerning Hypersignal is contained in the “HS_REV.TXT" and “ADDENDUM.TXT” file, found on Hypersignal directory.

In this tutorial, from time to time you will find paragraphs marked “POWER NOTES”. These notes are intended as extra, more advanced information for users familiar with both the DOS operating system and Hypersignal.

Please note that there are two basic forms of Hypersignal: non-real-time and real-time. The difference between the two versions is essentially real-time support for a variety of DSP/Acquisition hardware. Since the real-time software is a superset of the non-real-time software, all processing and display functions and commands not having to do with DSP/Acquisition hardware work the same for both versions. Hardware-related functions that are active only in the real-time software are so noted throughout this tutorial. See sections 1.1 and 1.2 in the Users Manual, Summary of Capability and Real-Time Version Capability, respectively.

*this page intentionally left blank*

# 2.0 GETTING STARTED

## 2.1.1 Installing Hypersignal from CD

To install Hypersignal software from CD, do the following:

1) Insert Signalogic software CD into CD-ROM drive

Win 9x or WinNT: From Start Menu select “Run”, and enter D:\SETUP where D denotes your CD-ROM. Click OK.

Win 3.1x: From Windows program manager select File|Run, and enter D:\SETUP as appropriate. Click OK.

2) Follow instructions shown by setup software.

After starting the setup software, you should see a box indicating which software items will be installed. In this box, you should see at least one of Hypersignal-Macro, Hypersignal-Macro Ex, or Hypersignal-Acoustic software selected, and possibly additional DirectDSP, Real-Time Composer, Visual Bench or DSPower software items selected as well.

## 2.1.2 Installing Hypersignal from Floppy Disk

**The instructions in section 2.1.2 below, Installing Hypersignal from Floppy Disk, assume that you are installing the software onto a hard disk. The standard versions of Hypersignal occupy too much disk space to be used in a floppy-disk-only configuration. There is an “academic version” of the Hypersignal non-real-time software available that can be used in a floppy-disk-only system. If you are using this version in a floppy-only system, insert distribution diskette #1 into a 1.2 Mbyte floppy drive, log to that drive, and skip to section 2.2 below, Running and Initializing Hypersignal.**

Below is the sequence for installing Hypersignal software from floppy disk:

1) copy *.* from distribution diskettes #1 and #2 to a subdirectory named "HSIGNAL" (or similar)

2) copy *.* from distribution diskette #3 and #4 to a subdirectory named "WAVEFORM"

Please refer to your operating system manual for information concerning creating subdirectories and copying files.

## 2.2 Running and Initializing Hypersignal

Although you can directly run Hypersignal by invoking the appropriate "HSxxxxx.EXE" file (where xxxxx is MACRO, ACOUS, MACNRT, etc. depending on the version of software that you have), we suggest that you use the batch file we have provided. The batch file first installs the "Metagraphics graphics driver" (makes it memory-resident), runs Hypersignal, and then removes the graphics driver from memory when you exit Hypersignal. This will keep the graphics driver out of your system memory when you are not using Hypersignal. Below is the basic sequence for running Hypersignal (note that steps 2 and 3 are not required if you've already run the software at least once, and performed an "Exit,Save" function when you exited the software):

***1) Invoking the initial batch file***

From Windows, select the "HS-Macro" icon and double click.

From command line, path to the "HSIGNAL" subdirectory (or whatever name you used when installing the software—see previous section 2.1, Installing Hypersignal, and run one of the following batch files (which one you have depends on your software version):

| | |
|---|---|
| HS <ENTER> | Hypersignal-Macro Non-Real-Time (NRT) |
| HSM <ENTER> | Hypersignal-Macro Real-Time |
| HSA <ENTER> | Hypersignal-Acoustic |
| HSN <ENTER> | Hypersignal-Macro NRT, Nihon version |
| HSMN <ENTER> | Hypersignal-Macro, Nihon version |
| HSAN <ENTER> | Hypersignal-Acoustic, Nihon version |

**NOTE: When you run Hypersignal for the first time, you will see a message of the form "HSxxxxx.CNF file not found — initializing new configuration...". All this means is that you must set up mouse and graphics driver selections and other basic information for the first time before the software is fully operational. When you "exit" Hypersignal, all selections and configuration will then be saved. When you next run the software, these set-up requests will not appear.**

***2) Select a graphics driver***

When you first run Hypersignal, a list of possible graphics driver selections will appear, and the software will not advance until a choice is made. Generally, when selecting a graphics driver, using the "best-guess probe" value displayed by the software (at top of screen) will produce reasonable results—better resolution may be possible if you know exactly what kind of video board is in your system. See Users Manual section 2.4.1, Selecting a Graphics Driver, for full discussion.

You might try an 800x600 or 1024x768 mode if one appears on the graphics selection page that matches your hardware. *If higher resolution modes don't work, you may not have enough video card memory—a safe bet is usually the standard* IBM VGA 640x480 *and* IBM EGA 640x350 *mode selections.* Another option that often works is selection 95 or VESA 800 x 600. Of course, if you know for certain that you have Hercules, AT&T, Compaq portable, or some other type of graphics, then you must go with this. Please note that there are four pages of graphics hardware selections—use the PgUp and PgDn keys if you don't immediately see your hardware listed (selections are in alphabetical order). Below is an example screen from the graphics driver selection list:

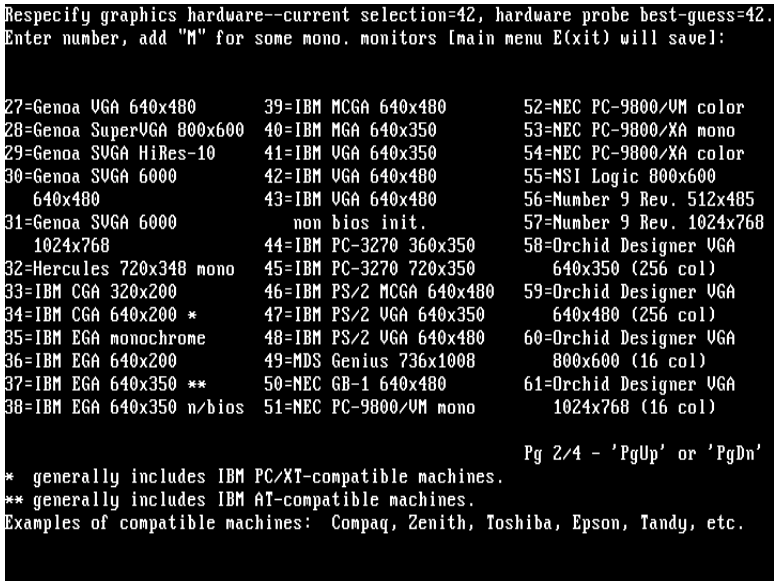
Respecify graphics hardware--current selection=42, hardware probe best-guess=42.
Enter number, add "M" for some mono. monitors [main menu E(xit) will save]:
27=Genoa VGA 640x480
28=Genoa SuperVGA 800x600
29=Genoa SVGA HiRes-10
30=Genoa SVGA 6000 640x480
31=Genoa SVGA 6000 1024x768
32=Hercules 720x348 mono
33=IBM CGA 320x200
34=IBM CGA 640x200 *
35=IBM EGA monochrome
36=IBM EGA 640x200
37=IBM EGA 640x350 **
38=IBM EGA 640x350 n/bios
39=IBM MCGA 640x480
40=IBM MGA 640x350
41=IBM VGA 640x350
42=IBM VGA 640x480
43=IBM VGA 640x480 non bios init.
44=IBM PC-3270 360x350
45=IBM PC-3270 720x350
46=IBM PS/2 MCGA 640x480
47=IBM PS/2 VGA 640x350
48=IBM PS/2 VGA 640x480
49=MDS Genius 736x1008
50=NEC GB-1 640x480
51=NEC PC-9800/VM mono
52=NEC PC-9800/VM color
53=NEC PC-9800/XA mono
54=NEC PC-9800/XA color
55=NSI Logic 800x600
56=Number 9 Rev. 512x485
57=Number 9 Rev. 1024x768
58=Orchid Designer VGA 640x350 (256 col)
59=Orchid Designer VGA 640x480 (256 col)
60=Orchid Designer VGA 800x600 (16 col)
61=Orchid Designer VGA 1024x768 (16 col)
Pg 2/4 - 'PgUp' or 'PgDn'
* generally includes IBM PC/XT-compatible machines.
** generally includes IBM AT-compatible machines.
Examples of compatible machines: Compaq, Zenith, Toshiba, Epson, Tandy, etc.

*Figure 2-1: Page 2 from the Graphics Driver List*

If you have already run Hypersignal, and “exited” (‘E’ key from main menu) when you were done, then a configuration file will exist with your current graphics driver selection saved inside, and the software won’t prompt you for new settings. If you wish to change the graphics setting, use the Graphics/Mouse Config menu selection under the Utilities column on the main menu.

***3) Select a mouse driver***

When you first run Hypersignal, a list of possible mouse driver selections will appear, and the software will not advance until a choice is made. A list of mouse drivers is presented, similar to the graphics driver selection process, above. When selecting a mouse driver, the “best-guess probe” value provided by the software may or may not produce reasonable results. The probe accurately detects the presence of memory-resident MOUSE.COM drivers and some mice connected directly to the COM1 port, but has trouble with other mice and with COM2 port connections. Best results are almost always obtained if you have previously installed the MOUSE.COM driver in your system memory, and then enter “5" for the memory-resident driver selection. This method allows the mouse manufacturer’s own driver, which should be the best driver in existence

for that mouse, to do most of the work. If a MOUSE.COM driver has not been installed, then good results are also usually obtained with Microsoft or Logitech mice connected directly to the COM1 port. In this case, enter '4C1' or '2C1', respectively. See Users Manual section 2.4.2, Selecting a Mouse Driver, for a full discussion.

If you have no mouse hardware in your system, then enter '7' for the keyboard-only selection. If you don't make this entry, then the software will continue to display an initial message of the form "no mouse hardware found..." every time it executes.

If you have already run Hypersignal, and "exited" ('E' key from main menu) when you were done, then a configuration file will exist with your current mouse driver selection saved inside, and the software won't prompt you for new settings. If you wish to change the mouse setting, use the Graphics/Mouse Config menu selection under the Utilities column on the main menu.

***4) Selecting a math coprocessor***

If you have an older 486 PC, where math processor was optional, this is done automatically, each time the software executes; no menu, list or set-up prompt is presented. When you run Hypersignal, the software automatically looks for an 80x87 math coprocessor, and uses it if found. If for some reason you later think that Hypersignal didn't do this correctly, you can override the automatic detection—see Users Manual section 2.3, Using a Math Coprocessor, for detailed information.

## 2.3 Configuring Hypersignal

**NOTE: This section assumes that Hypersignal is now running on your system, has been initialized with graphics and mouse driver selections, and that the "main menu" is displayed. If you have not already made graphics driver and mouse driver selections, please see the previous section 2.2, Running and Initializing Hypersignal. The following configuration steps assume that you know how to make basic menu selections and enter data into menu fields. If not, please take a quick look at section 3.2, Data-Entry Menus, or Users Manual section 3.1, Keyboard and Mouse Interface.**

### *1) Configure the the User Setup area*

Select the User Setup menu item under the Utilities column. The drive field should already be set for you. The way the display should look after the User Setup menu item has been selected is shown below:

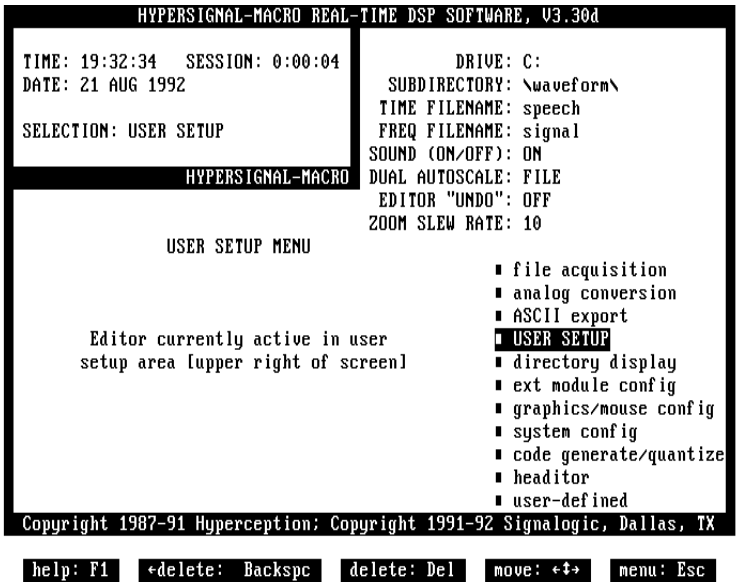


*Figure 2-2: After Selecting the User Setup Menu*

The subdirectory field will be set to whatever was current when you first ran Hypersignal—you will want to change this to be your "WAVEFORM" subdirectory (or whatever name you used for diskette #3 files when installing the software). The TIME FILENAME and FREQ FILENAME fields don't really matter, since you can enter these filenames in function and display data-entry menus as needed anyway. If you don't like programs beeping and buzzing at you when you do something wrong, then set the SOUND field to 'OFF'. Everything else should be left at the default. Pressing the 'Esc' key or pressing the 'Enter' key (or pushing any mouse button) while the cursor is on the last field will leave this function.

### *2) Initialize the System Configuration menu*

For both real-time and non-real-time software, enter the DEVELOPMENT DSP field in the System Configuration menu selection (last field, at the bottom of the menu), an example of which is shown below:

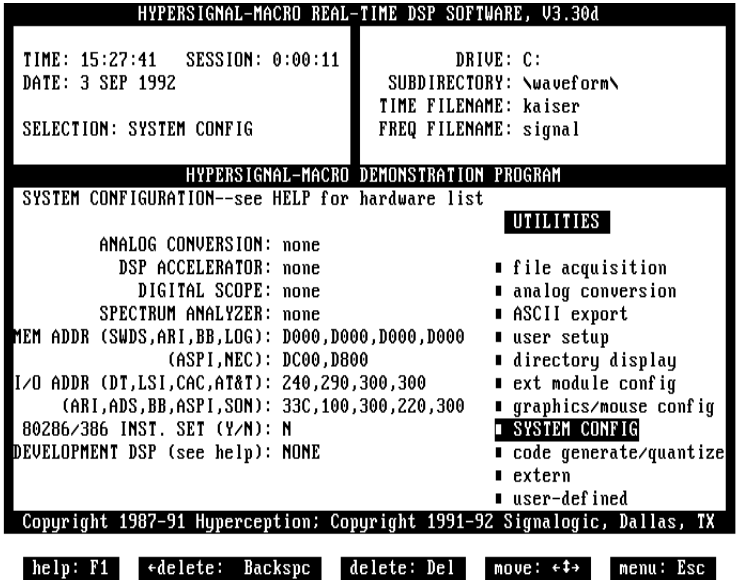


*Figure 2-3: System Configuration Menu*

Valid entries for the DEVELOPMENT DSP field include:

| | |
|---|---|
| 'AD21'<br>'AD60' | Analog Devices ADSP-210x, 2106x |
| 'WE32'<br>'WE32C' | AT&T DSP32 and DSP32C |
| 'M56'<br>'M96' | Motorola DSP56xxx and DSP9600x |
| 'N20' | NEC 77220 |
| 'TI10'<br>'TI20'<br>'TI25'<br>'TI30'<br>'TI40'<br>'TI50'<br>'TI54'<br>'TI62' | Texas Instruments TMS320C1x, TMS32020, C25, and TMS320C3x, C4x, C5x, C54xx, C62xx |

This field controls subsequent code generation and file acquisition from DSP manufacturer development tool output files. Entering this field is not required for the software to run—in fact, leaving it as a 'NONE' entry may

be useful for DSP chip-independent experimentation as you design digital filters—see Users Manual section 4.9.5, Generating DSP Code for Filters.

*If you have a real-time version of software*, enter the designator for your DSP/Acquisition hardware in the first four fields. You will need to select appropriate designators to replace the default 'none' entries that appear in Figure 2-3. The on-line Help for this menu item contains a full list of hardware designators. Selecting DSP/Acquisition hardware is only necessary to run those Hypersignal functions requiring external analog input (the ANALOG CONVERSION, DIGITAL OSCILLOSCOPE, and SPECTRUM ANALYZER fields). Functions that can use the DSP as an accelerator (the DSP ACCELERATOR field) will still operate using the host processor until DSP/Acquisition hardware has been selected.

See Users Manual section 2.5.2, Initializing the System Configuration Menu, for a full discussion.

**POWER NOTE 1:** **There are only two cases where you would not have the first four fields all the same: you have two or more supported boards in the system, or you purposefully want to disable the DSP ACCELERATOR, in which case a selection of "none" will cause the 80x86/80x87 to be used instead. A situation for the latter would be for analog hardware that has no DSP chip, in which case either 'none' or some other DSP board selection would have to be entered for DSP ACCELERATOR. If you are using an analog-only board with no DSP chip, or no D/A playback capability, then whatever board you enter in the the DSP ACCELERATOR field will assume those responsibilities. For example, if you are using an analog-only board for your SPECTRUM ANALYZER selection, then you must have a valid DSP ACCELERATOR selection working in order to run the spectrum analyzer, since the real-time Fourier analysis can't be done on your analog input board.**

The MEMORY BASE ADDRESS and I/O BASE ADDRESS fields are somewhat complicated. Normally, you will want to leave these two fields at their default. However, there can be times when you need to change base addresses—the best examples being a possible conflict with boards you already have in your system, or a manufacturer that ships different boards with different base addresses. Note that there is one entry for every

manufacturer we support—not for every board. If you need to change a base address, *be sure you only change the value for the relevant manufacturer*--the other values do not need to be correct, but *they must remain entered!* Enter zeros for the other manufacturers if you must—but enter only one value for every manufacturer listed in the field label. You can always get the defaults back if necessary by deleting the "HSMACRO.CNF" configuration file (see section 3.4, Saving the User Environment). If you have two or more boards from the same manufacturer, then Hypersignal will consider the first board mentioned in the four "operation" fields (ANALOG CONVERSION, DSP ACCELERATOR, DIGITAL OSCILLOSCOPE, and SPECTRUM ANALYZER) to be at the entered I/O base address, the second board to appear in the operation fields to be at the I/O base address plus 10H (16), the third board at the I/O base address plus 20H, and so on.

**NOTE 1: All memory and I/O base addresses must be entered in hex.**

**NOTE 2: Check the Hardware Reference Guide for exact base address values for specific hardware.**

The 80286/80386 INSTRUCTION SET field should generally be set to 'Y' if you have an 80386 or better machine, including Pentium. Depending on the board design and data transfer scheme implemented by the board manufacturer, this field may or may not matter. Only change it to 'N' if you definitely have a legacy 8086 or 8088 (IBM PC/XT machine) or you run into problems later, such as "spikes" in your data, or board freeze-ups during real-time operation. See the Hardware Reference Guide for exact settings for specific hardware.

***3) Initialize the External Program Configuration menu***

Although not required in any way for the software to operate fully, you might like to make one or two external program selections before continuing. This is a completely optional configuration step, and can be performed later as needed.

For example, you might want to set up a text-editor as a menu item under the Utilities column, so that you can go directly to your editor from Hypersignal to review generated source code, or set up a debugger so that you can see what Hypersignal has downloaded and is executing on your

DSP board. To do this, select the Ext Program Config menu item. In the appropriate field for where you want the external program menu label to appear on the Hypersignal main menu, enter the filename of your program, followed by a comma, followed by the menu label as you would like it to appear. For instance, an editor might be set up by entering

editor,text-editor

in the UTIL FILE, MENU LABEL 1 field. Such an entry would cause Hypersignal to look for an executable file of name "editor.com" first, then "editor. exe", then "editor. bat", in that order, whenever the menu item "TEXT-EDITOR" is selected from the Hypersignal main menu. Of course, you can explicitly enter an extension if not ".COM", ".EXE", or ".BAT". Be sure to specify where the external file is located in the DRIVE and SUBDIRECTORY fields, although Hypersignal will look elsewhere, including the system path list, for the file if not found immediately. Pressing the 'Esc' key or pressing the 'Enter' key (or pushing any mouse button) while the cursor is on the last field will leave this function. See section 3.6, Invoking External Programs, for more information.

***4) Save your current environment***

Press the 'Esc' key (or leftmost two buttons on mouse) until you are looking at the main menu, with no data-entry menus displayed. Now, press the 'U' (update) key to *save all current driver selections, system configuration, user setup, data entry menu fields, etc.*, in the user-environment configuration save file. The Update function saves your current environment and stays in the main menu. The Exit function ('E' key) saves your current environment and exits Hypersignal.

Updating the configuration file is intended to save your current environment between Hypersignal sessions, and also to avoid unnecessary repetition of basic setup and configuration tasks.

***5) Use the software***

At this point, you should be up and running! You can now run Hypersignal functions by entering filename fields with the example waveform filenames included on the distribution diskettes—some example filenames are upchirp, speech, elliptic, and signal. See the next chapter, USING HYPERSIGNAL, for some brief suggestions on how to get around and invoke functions and displays. See subsequent chapters for specific, step-by-step Hypersignal examples.

# 3.0 USING HYPERSIGNAL

Now you should be “up and running”, looking at the “main menu”. Three main categories of functions are available: Time Domain, Frequency Domain, and Utilities. To get around, move the mouse or use the arrow keys on the keyboard. If your mouse feels too fast or too slow, see Users Manual section 2.4.2.1, Mouse Driver Trouble Shooting, and Software Reference Guide section, 3.4, User Setup Reference (ZOOM SLEW field), for suggestions. To select a menu item, push any button on the mouse or the ‘Enter’ key on the keyboard. An example main menu display is shown below:

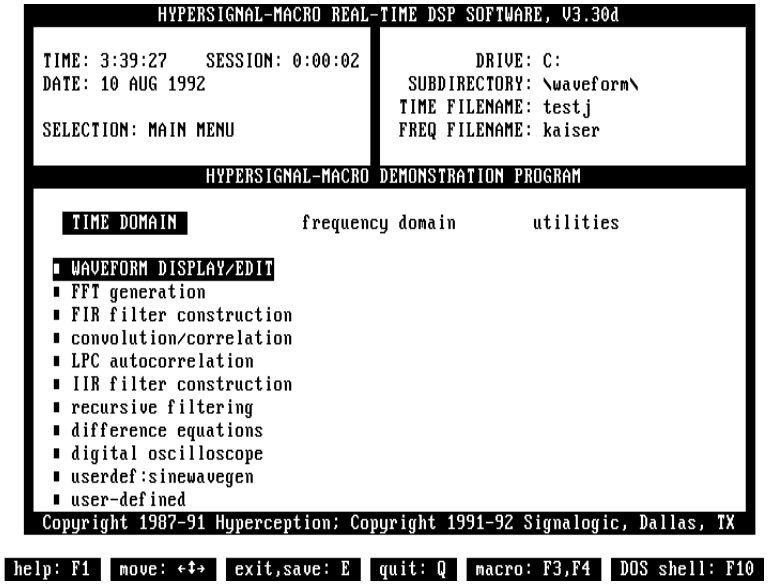


*Figure 3-1: Hypersignal Main Menu*

To back up from wherever you go, always use the ‘Esc’ key (or the leftmost two buttons on a three-button mouse). See Users Manual section 3.1, Keyboard and Mouse Interface, for a complete description of keyboard and mouse commands.

## 3.1 On-Line Help

To see on-line Help, press the ‘F1’ or ‘?’ keys on the keyboard (or the rightmost two buttons on a three-button mouse). Below is an example of an on-line Help display brought up on top of a data-entry menu:

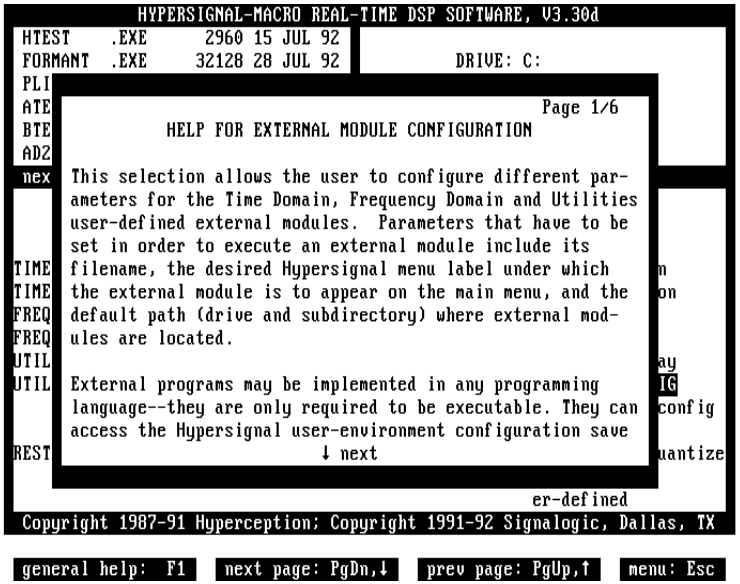
HYPERSIGNAL-MACRO REAL-TIME DSP SOFTWARE, V3.30d
HTEST .EXE 2960 15 JUL 92
FORMANT .EXE 32128 28 JUL 92
DRIVE: C:
Page 1/6
HELP FOR EXTERNAL MODULE CONFIGURATION
This selection allows the user to configure different parameters for the Time Domain, Frequency Domain and Utilities user-defined external modules. Parameters that have to be set in order to execute an external module include its filename, the desired Hypersignal menu label under which the external module is to appear on the main menu, and the default path (drive and subdirectory) where external modules are located.
External programs may be implemented in any programming language--they are only required to be executable. They can access the Hypersignal user-environment configuration save
↓ next
Copyright 1987-91 Hyperception; Copyright 1991-92 Signalogic, Dallas, TX
general help: F1
next page: PgDn,↓
prev page: PgUp,↑
menu: Esc

*Figure 3-2: Example of On-Line Help Display*

You can get on-line Help at any time in the program, no matter where you are. Once inside an on-line Help display, move the mouse up and down or use the up and down arrow keys on the keyboard to browse through the on-line Help information. The left button on the mouse (‘Home’ key on keyboard) jumps to the beginning of the on-line Help, and the right button (‘End’ key on keyboard) jumps to the end. See Users Manual section 3.2, On-Line Help, for detailed information.

## 3.2 Data-Entry Menus

After selecting an item from the main menu, you will see a “data-entry menu” that contains one or more fields that specify input, operating controls, and output, for the chosen function or display. When entering data into these menus, here are the basic rules:

Use the ‘Enter’ key (or push any button on the mouse) to enter a field and go to the next field.

Use the up arrow key (or move the mouse up) to back up to previous fields.

Use the ‘Delete’ or ‘Backspace’ keys to erase data or blank a field.

**NOTE: If you are already in a data-entry menu, and are anxious to “do the function” or “see the display”, there are two ways to advance to actual function processing or graphical display: 1) entering the last field (pressing the ‘Enter’ key or any mouse button when the cursor is on the last field in the menu), or 2) pressing the <ctrl-Enter> key sequence (use <ctrl-J> key sequence on NEC PC-980x machines). Doing either of these will cause function processing or display to begin, as appropriate.**

Use the ‘F1’ key (or rightmost two buttons on a three-button mouse) to get on-line Help at any time. The ‘Esc’ key from on-line Help will return immediately to your current editing position.

***Use the ‘Esc’ key (or leftmost two buttons on a three-button mouse) to leave the data entry menu and return to the main menu.***

Waveform filenames are used to specify input and output (if needed) for most displays and functions. For displays and functions that require analog input, reserved waveform names are used (such as ‘a0’ for analog channel 0). In either case, input and output is usually specified in the first two fields, or “filename fields”, of data entry menus. For filename field entry, the ‘Enter’ key will truncate the remainder of the current field past the cursor position; for non-filename fields, truncation does not usually occur. If you are just changing a single letter in a filename field, or are making some other minor change, use the down arrow key (or move the mouse down) to avoid truncation. If an entered filename can’t be located, a “pop-up waveform directory” will appear (see section 3.2.2, Pop-Up Waveform Directories).

See Users Manual section 3.4, Using the Hypersignal Data-Entry Editor, for complete, detailed information about data-entry menus.

### 3.2.1 Waveform Summaries

Waveform summaries are displayed in the System Status area (upper left of main menu) after successful loading of waveform file(s) for display or processing. A waveform summary displays parameters and attributes from the waveform file header. For detailed information on these values, see Users Manual section 3.4.1, Waveform Summary Display.

To redisplay a waveform summary that has been overwritten by pop-up directory display information or status messages, you can reload the waveform by repositioning the cursor at the beginning of the waveform filename field and pressing the ‘Enter’ (carriage return) key or any mouse button.

In the case of dual-waveform display functions, the waveform summary always applies to FILENAME1.

Below is an example of a waveform display summary:

```
          Number of samples: 1015672
Type/complex/numchn: magH[n]/real/1
        Max |amplitude|: 19103.0
    Analysis framesize: 4096
             FFT order: 12
    Sampling frequency: 44100Hz
```

Especially note the “Type/complex/numchn” descriptor, which, in this example, indicates a non-complex, single-channel, filter (transfer function, or “H[n]”) magnitude (frequency domain) waveform file. The “type” of a waveform actually consists of two parts: the type of data, and its original source.

### 3.2.2 Pop-Up Waveform Directories

Pop-up directory display appears automatically during data-entry menu operation in the upper left of the display when an entered filename cannot be found. Also, you can manually invoke pop-up directory display from inside any data-entry menu by using the ‘PgUp’ key.

Pop-up directory display includes waveform filename and extension, length, and date of creation. The search strategy for different waveform extensions and/or more specific waveform filename may be changed via consecutive use of the ‘PgUp’ key (see Users Manual section 3.4.2.1, Pop-Up Directory Waveform File Search Strategy).

See Users Manual section 3.4.2, Pop-Up Directory Display, for more information.

## 3.3 Using the Printer

To print Hypersignal menu displays, you should use the standard DOS <shift-Print Scrn> key sequence; there is no built-in Hypersignal function for printing menu displays.

To print waveform displays, both to printers and disk files, you should use the built-in Hypersignal function that can perform both “screen dumps” and “publication-quality” prints. A screen dump is defined as simply transferring the current contents of video RAM directly to the printer, without any increase in resolution. Of course, for printers with resolution greater than about 90 dots per inch, this will not fully utilize the capability of the printer. A publication-quality print is defined as regenerating the current screen display in EMS memory, at much higher resolution, and then transferring this image to the printer.

The built-in waveform print function is invoked by the ‘F2’ key inside all waveform displays. Below is an example waveform display with the pop-up print menu shown:

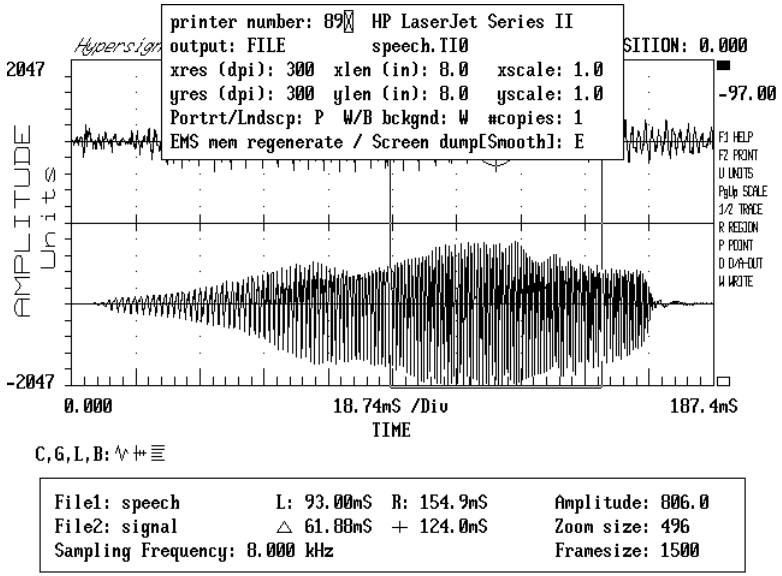


*Figure 3-3: Example of Pop-Up Print Menu*

See section 5.3, Hardcopy Print-Out, for a step-by-step example of using the built-in print function. See also Users Manual sections 3.5, Printing Menus and Waveform Displays, and 4.3.2, Printing Waveform Displays, for full details on using the built-in print function.

## 3.4 Saving the User Environment

As mentioned in sections 2.2 and 2.3, Running and Initializing Hypersignal, and Configuring Hypersignal, when first invoked, Hypersignal looks for a "configuration file" saved from a previous Hypersignal session (all current program settings are stored in this file).

The "user-environment configuration file" is an ASCII text file that contains the complete state of all user-controlled parameters in Hypersignal. This file is intended to be saved between Hypersignal sessions so that frequent use of the software is as convenient as possible, and no time is spent rethinking set-up or initialization parameters. The configuration file is saved on disk under the name "HSMACRO.CNF" for all versions of Hypersignal software.

## 3.4 Saving the User Environment

The user-environment configuration file includes the current graphics and mouse hardware drivers, current System Configuration, currently selected acquisition/DSP hardware (real-time software only), current User Setup area, all current menu selections, all current user-modifiable data (filenames, analysis parameters, display options, etc.), current editing cursor position, and so forth. Every user-modifiable parameter in Hypersignal is saved in the configuration file.

When running Hypersignal, the user-environment configuration file can be saved in three ways:

1) By "exiting" from the main menu ('E' key). Exiting the software saves the configuration file and stops Hypersignal execution.

2) By "updating" from the main menu ('U' key). Updating saves the configuration file but Hypersignal continues to execute.

3) By executing an external program or DOS shell. The configuration file is saved automatically by the program in anticipation of its being accessed by the external program.

**NOTE: If you "quit" from the Hypersignal main menu, the user-environment configuration file is not saved.**

**POWER NOTE 2: You can invoke Hypersignal from other subdirectories. Generally speaking, Hypersignal searches for system files, HELP file, and printer-definition file first on its own subdirectory, then on the currently logged DOS path, and then in the DOS path list. The HELP and printer-definition files are also searched for on the USER SETUP path found in the configuration file.**

**POWER NOTE 3: The current DOS path (drive and subdirectory) is the first place where Hypersignal expects to find the configuration file. Moreover, this is *always* the place where Hypersignal saves its configuration file. If Hypersignal cannot locate a configuration file on the current DOS path, it then searches the path containing the executable file ("HSxxxxx.EXE") and the DOS path list. If a compatible configuration file is found, then the software will start with this file, but save all new settings in a new file on the current path. Note that this provides the means to set up a multiuser system with multiple configuration files: the initial configuration file can be provided from a master path, and individual, personalized configuration files can be saved to local paths.**

## 3.5 Invoking a DOS Shell

To go temporarily to a "DOS shell", where you can enter DOS commands, use the 'F10' key. Type "exit" when you are done with DOS to return directly to Hypersignal. See Users Manual section 3.6, Invoking External Programs and the DOS "Hot-Key", for detailed information.

## 3.6 Invoking External Programs

From the Hypersignal main menu, you can execute external programs, and also invoke a DOS "shell" (actually an executable program in the form of the DOS "COMMAND.COM" file).

On the main menu, "User-Defined" menu labels are available (two each under the Time Domain, Frequency Domain, and Utilities columns).

Also, the F10 key has been installed in the main menu as a DOS "Hot-Key", which allows you to jump to a DOS shell at any time from the main menu, perform DOS operations, and "exit" quickly back to Hypersignal.

Use the Ext Program Config menu selection to set up your external program filenames and desired Hypersignal menu labels. For example, a typical External Program Configuration menu might look like this:

```
TIME FILE,MENU LABEL1: sinegen.exe,sinegen
TIME FILE,MENU LABEL2: prog1.com,menu_lab_1
FREQ FILE,MENU LABEL1: prog2.exe,menu_lab_2
```

```
FREQ FILE,MENU LABEL2: prog3.bat,menu_lab_3
FREQ FILE,MENU LABEL3: prog4.xxx,menu_lab_4
 UTIL FILE,MENU LABEL1: edit.com,text-editor
 UTIL FILE,MENU LABEL2: prog5.yyy,menu_lab_5
                DRIVE: C:
          SUBDIRECTORY: HSIGNAL
RESTORE DOS PATH Y/N?: Y
```

where “sinegen.exe” is an example external program provided on the distribution diskettes (intended as an example of how to access the Hypersignal user-environment configuration file), “edit.com” is a prototypical text editor, and “progN.ext” are prototypical user-defined programs.

To then execute, for example, the text editor set up in Utilities menu label 1, above, you would choose from the main menu the “Text-Editor” menu selection, under the Utilities column.

The DRIVE and SUBDIRECTORY fields specify the location of external programs; the RESTORE DOS PATH determines whether the software should return the DOS path to its initial setting after the external program or DOS shell executes and returns control.

See Users Manual section 2.5.3, Initializing the External Program Configuration Menu, for more detailed information on the External Program Configuration menu, and the process of setting up external programs and defining main menu labels.

*this page intentionally left blank*

# 4.0 ACQUIRING DATA

What good is a DSP program if you can't get some data into it? Discussed first in this chapter is the File Acquisition function, which functions identically in all versions of Hypersignal; discussed second is the Analog Conversion function, which requires an DSP/analog board, and applies only to real-time versions of Hypersignal.

## 4.1 Acquiring Data From Disk Files

First, let's briefly use the File Acquisition function. Since you may or may not have a data file of your own available to acquire, we'll just "re-acquire" an existing Hypersignal waveform. Enter the following File Acquisition menu:

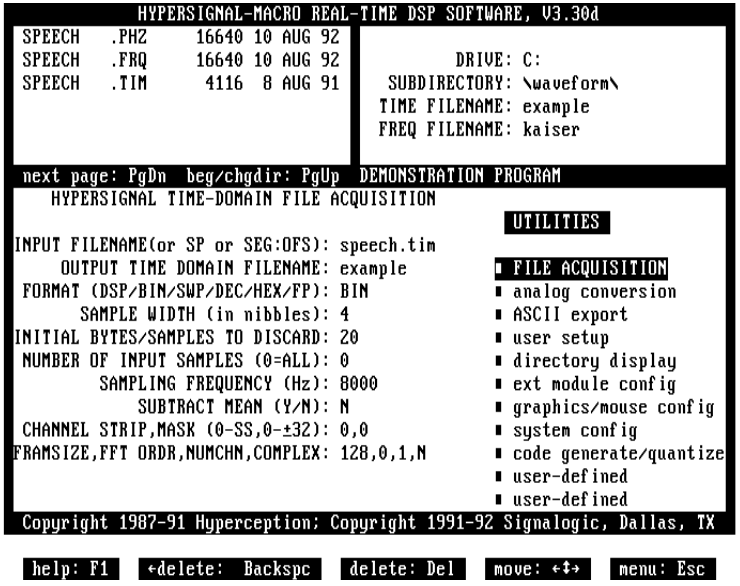


*Figure 4-1: File Acquisition Menu*

Pressing the 'Enter' key (or any mouse button) while the cursor is positioned on the last field will cause the existing time waveform file "speech" (included on waveform diskette #3) to be imported into Hypersignal, and

a new time waveform file, “example”, to be created. Note that the extension for the input file is required, since the input could be any file, from any source. Note also that no extension should be entered on the output file—this is true of *all* Hypersignal functions: once a waveform is in Hypersignal format, the program expects to add extensions automatically, on its own. In some cases you can override this, but generally you won’t need to.

What are these other parameters? Well, the ‘BIN’ entry for FORMAT tells the program to expect binary input data. ‘DEC’, ‘HEX’ and ‘FP’ are all ASCII formats; in these cases Hypersignal would expect ASCII decimal, hexadecimal, and floating-point data, respectively. A ‘DSP’ entry would refer to the “Development DSP” field in the System Configuration menu (see section 2.3, Configuring Hypersignal), instructing the program to automatically search for files in the formats used by the manufacturer of the “Development DSP” chip.

A SAMPLE WIDTH entry of 4 tells the program that each sample in the input data is to be 16 bits wide (entry is in units of “nibbles”, or 4-bit chunks).

INITIAL BYTES/SAMPLES TO DISCARD indicates the number of bytes (for binary formats BIN and SWP) or samples (for all other ASCII formats) to discard from the beginning of the input file. This can be useful if the file you are acquiring has a header put there by another program. Note that in the case of the example, we are discarding the 10-word (20-byte) Hypersignal “short header” on the “speech.tim” file. We do this because otherwise the header data would be acquired as actual sample data, which would then appear as garbage at the beginning of the waveform. The File Acquisition function attaches a new header to all input files based on what you tell it in this menu, and what it learns about the data on its own.

**POWER NOTE: Hypersignal waveform file headers are either short or extended. See Users Manual chapter 5, HYPERSIGNAL WAVEFORM FILE FORMAT, for detailed information about waveform file headers.**

NUMBER OF INPUT SAMPLES is very straightforward—we have entered a ‘0’ (zero), which tells the program to get as many samples as the in-

put file is long. If you want a lesser number, or need to avoid the end of a file for some reason, enter a specific value here.

If the SUBTRACT MEAN filed is entered with a 'Y', any dc offset in the input data will be removed. The calculated mean is exact for input waveforms less than 16384 samples; for waveforms longer than this, the mean becomes a running sum that represents the last 16384 samples at any point in the input data.

CHANNEL STRIP and MASK can be used to "strip" out individual channels from interleaved, multichannel input files, and also to "mask" bits at either end of individual samples, which can be useful for data with additional information encoded in each sample. In the case of the example, an entry of '0,0' specifies no stripping and no masking.

The POST-ACQUISITION FRAMESIZE, FFT ORDER, NUMBER OF CHANNELS, and COMPLEX subfields have no bearing on the actual acquisition process, but are only intended as information in the final file header that tells other Hypersignal functions how to treat this waveform. The NUMBER OF CHANNELS value is definitely important and should be entered correctly if you are acquiring a stereo or multichannel waveform, and you want Hypersignal to continue to treat the waveform as such. FRAMESIZE and FFT ORDER might be what you think convenient values for subsequent FFT, time display, etc. would be; however, since you can override these values at any time in the functions where they are needed, it is not that important what you enter here. In fact, entering '0' for these values will disable automatic initialization by subsequent functions using the acquired waveform. The COMPLEX subfield should be set if you are acquiring complex data—enabling this field will instruct Hypersignal to assume that the incoming data stream contains alternating real and imaginary values.

Some other examples of typical data import:

| **TYPE OF DATA** | FORMAT | SAMPLE WIDTH |
|---|---|---|
| **Unsigned 8-bit binary** | BINU | 2 |
| **32-bit ASCII hex** | HEX | 8 |
| **ASCII floating-point** | FP | don't care |

| | | |
|---|---|---|
| **unsigned ASCII floating-point** | FPU | # of nibbles equiva-lent to half of positiveinput range |
| **ASCII decimal integer** | DEC | 2 |

## 4.2 Acquiring Data From DSP/Analog Boards

Now, let's use the Analog Conversion function to get some live data from the DSP/Analog board.

> **NOTE: Remember, this is a real-rime software function, so if you have non-real-time software, please go to chapter 5, DISPLAYING/EDITING WAVEFORMS.**

We assume here that you have set the ANALOG CONVERSION field in the System Configuration menu to the board that is currently in your machine (see section 2.3, Configuring Hypersignal, if not). We also assume that you have a signal (function generator, or microphone & amplifier, etc.) connected to input channel 0 on your board, and a speaker connected to output channel 0. See the board's documentation to determine which connector(s) on the board's back plane are input and output channel 0.

Assuming that the System Configuration menu is accurate, and connections are in place to the analog input(s) to the board, enter the following Analog Conversion menu:

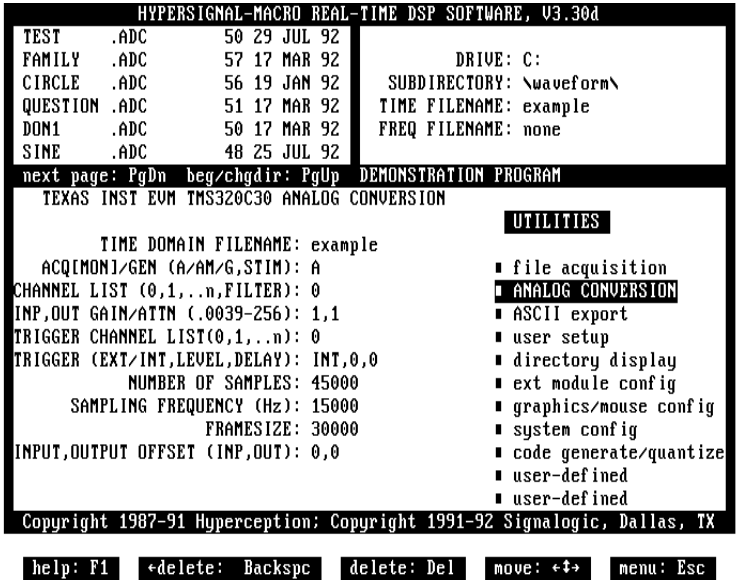


*Figure 4-2: Analog Conversion Menu*

Pressing the ‘Enter’ key (or any mouse button) while the cursor is positioned on the last field will cause any existing input files of the same name to be checked for, the input waveform to be prepared, and a final “BEGIN ACQUISITION?” prompt to appear. After answering ‘Y’ to this prompt, acquisition will occur for a period of three (3) seconds (45000 samples / 15000 samples/sec) and the result will be stored on disk as the time waveform file “example” (actual filename “example.tim”).

What about the parameters we have entered? First, the ‘A’ entry in the ACQ/GEN field instructs the program to acquire data; a ‘G’ would instruct it to generate continuous output data by reading the TIME DOMAIN FILE. An entry of ‘AM’ in this field would enable the “monitor mode” while acquiring data—this loops back the input channels on the CHANNEL LIST to the corresponding output channels. For example, if you were recording music, this would allow you to listen before and during the recording time. For any application, the monitor mode can enable you to use this function as a continuous real-time filter (see section 10.3, “Snap-In” Real-Time Filtering) by simply never answering the final “BEGIN ACQUISITION?” prompt.

The CHANNEL LIST field is what controls the number of channels you want in operation at one time. In this example, we asked for single-channel operation on channel 0. An entry of '0,1' would record both channels 0 and 1 simultaneously. For hardware with more than two channels, the '-' character can also be used to specify a range of channels: an entry of '0,1,4-7' would acquire channels 0, 1, and 4 through 7, creating a 6-track waveform. Note that you *should not adjust sampling frequency* depending on the number of channels—Hypersignal automatically programs the board to ensure that each channel is acquired at the rate entered, regardless of whether the board has true concurrent sampling or not. Note also that *when using a board with a DSP chip*, you can enter a filter name after the channel list. For example, an entry of '0, 1,elliptic' cause the filter file 'elliptic' (assumed to be previously designed in Hypersignal—see chapter 9, DESIGNING FILTERS) to be downloaded and applied in real-time to both channels 0 and 1.

INPUT and OUTPUT GAIN are digital scaling factors applied mathematically to the input or output data by the DSP chip. For boards that do not have a DSP chip, the only time these fields apply is when the board has programmable gain registers. If, for instance, you wanted to boost your input levels 5 times, then an entry of '5, 1' might be appropriate. Note that in the case of a DSP chip applying scaling factors mathematically, this is not a true "gain"; the resolution of the analog converter remains unchanged, the converter values being modified only after the conversion process (for input) or before the conversion process (for output).

The TRIGGER CHANNEL LIST field is active only when acquiring. This field tells the board which channel(s) must be the trigger criteria in order to start the acquisition process. Note that for some two-channel boards, an entry of '0,1' specifies that *both* channels 0 and 1 must meet the criteria. A CHANNEL LIST entry of '0' and a TRIGGER CHANNEL LIST entry of '1' would tell the program that data is to be acquired on channel 0, but only after channel 1 meets the trigger criteria. Note that for single-channel boards, CHANNEL LIST and TRIGGER CHANNEL LIST should be the same (zero).

The TRIGGER field specifies trigger criteria, and is also only active during acquisition operations. An entry of 'INT' means use software (threshold level); an entry of 'EXT' means use the board's external TTL-level trigger (if it has one). The value entered for LEVEL can be either positive or negative; the software will only trigger when input with the same sign exceeds the magnitude of LEVEL. Note that a LEVEL value of zero basi-

cally is “free-run”: the first input sample will trigger, since the magnitude of all possible input values is >= 0. Also note that LEVEL values should be entered in “A/D units”, not Volts (unless otherwise specified in the hardware reference documentation). Currently, the DELAY value is only used for “stimulus & response” operation (see chapter 12, STIMULUS & RESPONSE).

NUMBER OF SAMPLES is the amount of data that will be acquired/generated. Again, the program ensures that this amount of data is processed for each channel during multichannel operation.

SAMPLING FREQUENCY is the rate at which a waveform file is acquired/generated, i.e., the number of times per second the waveform voltage is sampled.

Similar to the File Acquisition function, the entry of FRAMESIZE here has no bearing on the actual acquisition/generation process, but is intended for post-acquisition use when doing further processing on acquired waveforms. We have entered a ‘0’ to disable automatic initialization by the program in any subsequent functions. For data for which you know what framesize you will be needing, entering this value here would make subsequent functions more convenient.

INPUT, OUTPUT OFFSET are either positive or negative digital offsets added mathematically to the input or output data by the DSP chip. For boards that do not have a DSP chip these fields still apply, but be careful: your maximum possible sampling rate may decrease, since any offset values other than zero will cause the 80x86 processor to have to do the work (and the 80x86 is considerably slower at this sort of thing than a DSP). If, for instance, you wanted to compensate for a dc offset in an A/D converter, then an entry of ‘567,0’ might be appropriate. Note that the value entered should be in actual “A/D units”, not Volts. Also note that if a scaling factor (see above) is also being applied, the scaling factor is always applied first, then the offset added afterwards. This is true for both input and output operation.

*this page intentionally left blank*

# 5.0 DISPLAYING / EDITING WAVEFORMS

In this chapter, several aspects of Hypersignal waveform display from an overall perspective, independent of type of data (time domain, frequency domain, or filter), are discussed. Some of the operations covered include display at zoom levels, panning from frame to frame, and changing units and scaling. Display in log mode is covered. Also, time domain editing in both “cut-and-paste” mode and “rubber-band” modes, and typical hardcopy print-outs from Hypersignal waveform displays are discussed. Finally, for real-time software versions, playback of waveform segments to the output D/A of the currently selected DSP/Analog board is discussed. In this chapter, we will walk through the Waveform Display/Edit function, a time domain display function, although many examples apply to waveform display in Hypersignal in general. First, enter the following Waveform Display/Edit menu:

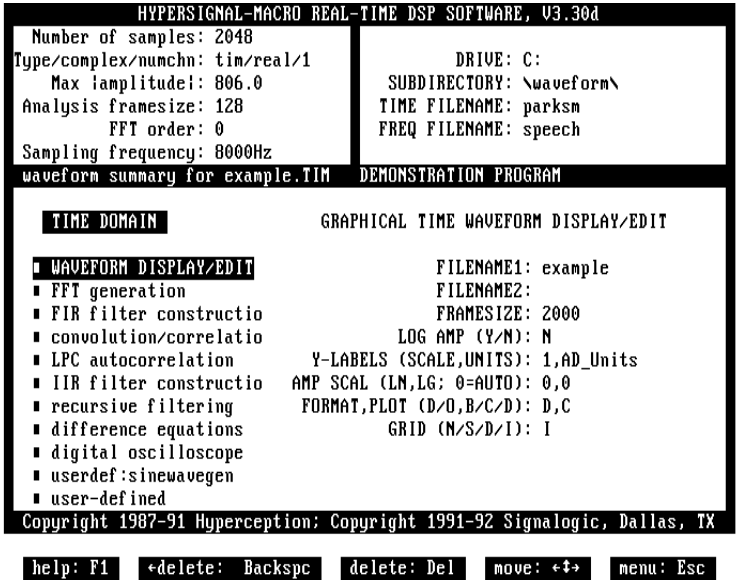


*Figure 5-1*

Remember to enter the last field or press <ctrl-Enter> to cause graphical

display to begin. The above menu will cause the program to utilize the full display area to show the “example” waveform, starting with the first 2000 samples. The LOG AMP field has been disabled (causing linear amplitude display). The Y-LABELS entry will cause no scaling to occur (scale factor=1), and a label of “AD_units” to be printed on the vertical axis. An AMP SCAL field entry of ‘0,0’ causes the program to use the maximum value found in the waveform as the maximum vertical axis value (autoscaling). If dual-trace display were active (which it is not in this example because the FILENAME2 field is blank), a FORMAT entry of ‘D’ would specify dual-dual trace display; an ‘O’ entry would specify overlaid-trace display. A PLOT entry of ‘C’ asks for “connected” waveform depiction, where individual samples are connected by a straight line (a ‘D’ entry would show “discrete” sample values, and a ‘B’ entry would show both). A GRID entry of ‘I’ asks for “intersected” grid display, in which a dot appears wherever tic marks on the horizontal and vertical axes would intersect (‘S’ indicates solid grid lines, ‘D’ indicates dashed lines, and ‘N’ disables grid display entirely).

Pressing the ‘Enter’ key (or any button on the mouse) while the cursor is positioned on the last field will advance to the actual graphical display screen, as shown below:

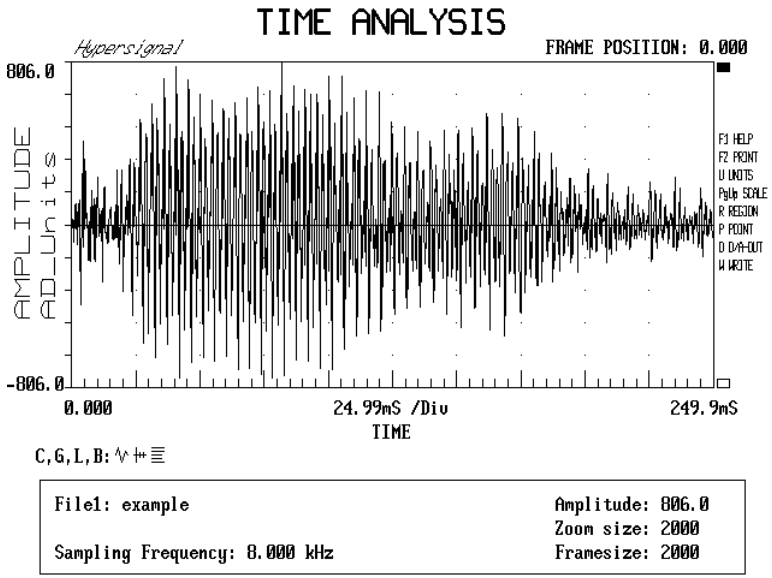


*Figure 5-2*

If you are using non-real-time software you should see the picture above, because “example” is merely the distribution diskette “speech” file after being re-acquired. If you are using real-time software, you should see data from your analog signal source.

## 5.1 Display, Zoom, Pan, and Scaling

Now use the arrow keys (or move the mouse) until the target window (“zoom box”) is positioned so that it occupies about 1/5 of the display area, and is all the way over to the left. The left and right arrow keys (mouse motion) move the zoom box from side to side; the up and down arrow keys (mouse motion) expand and contract the size of the zoom box.

Once positioned, press the ‘Home’ key (or middle button on the mouse) to zoom in on the waveform. The waveform area contained in the zoom box at this time will become the new display.

Repeat the zoom process a few times until it feels comfortable. Note that as you magnify data, individual samples become visible.

To back up to previous levels of zoom, use the outer two buttons on the mouse. A maximum of 16 levels of "zoom history" are stored by the program.

To pan to the next frame of data, use the space bar key (or righthand button on the mouse). To pan to the previous frame of data, use the 'Backspace' key (or lefthand button on the mouse). Note that the pan function is always active, regardless of the current level of zoom.

Try pressing the 'U' key a few times; this should toggle the current time units from their current SI value to 10e3 higher, then to 10e-3 lower, then to "point" readout (in samples), then back to the default setting.

To expand and contract data vertically, press the 'PgUp' and 'PgDn' keys, respectively.

**POWER NOTE: The Waveform Display/Edit function can be used to display any waveform, not just time waveforms. Enter the filename *plus extension* in the FILENAME1 and FILENAME2 fields to override the default .TIM extension.**

Now, press the 'Esc' key (or the left two buttons together on the mouse) and enter the following Waveform Display/Edit menu:

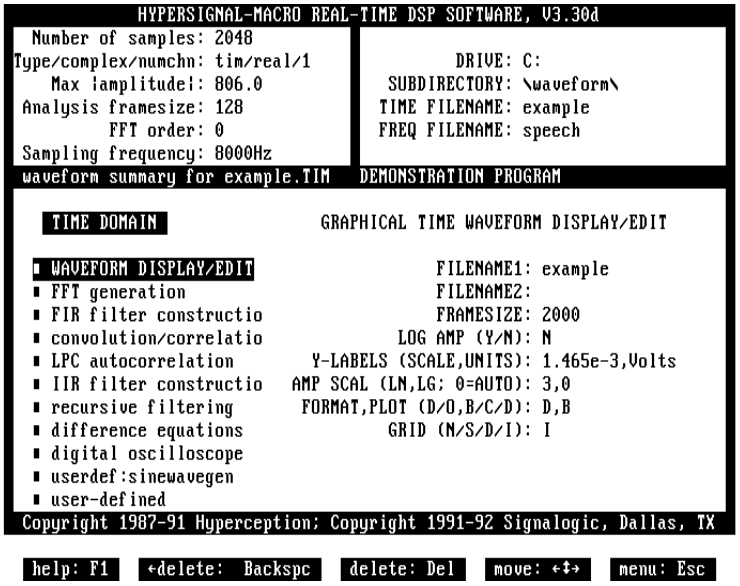


*Figure 5-3*

This menu is identical to our first one except that we have invoked some scaling and we have changed the data depiction style. The scale factor value comes from dividing the full-scale voltage of the source of the input data by the resolution of the source. For example, if an analog board was used that had 5 Volts full-scale and 16-bits of input, this would be 5/32767, or .1526e-3. For data obtained with 3 V full-scale and 12-bits resolution, the factor would be 3/2047, or 1.465e-3. In the above example we have accordingly changed the label that will be printed on the vertical display axis from “AD_units” to “Volts”. We have also changed the linear AMP SCAL value to be 3, since we are now working in Volts and don’t expect any data higher than 3 Volts. Of course, leaving this field set to ‘0’ would still tell the program to set the maximum vertical axis value to the maximum magnitude found in the waveform. The last change we made was to the PLOT field, which is now set for both “connected” and “discrete” data depiction, to better see individual samples.

Pressing the ‘Enter’ key (or any button on the mouse) while the cursor is positioned on the last field will advance to the actual graphical display screen.

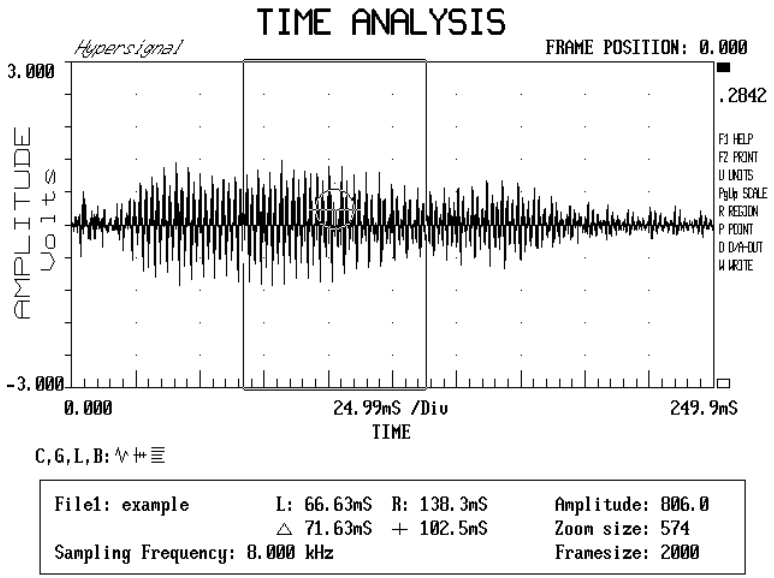


*Figure 5-4*

Do a few zooms so that you can see the “discrete” style of sample depiction clearly. After two 1/3 display-width zooms, and one ‘PgUp’ key, you should see a display similar to Figure 5-5, below:

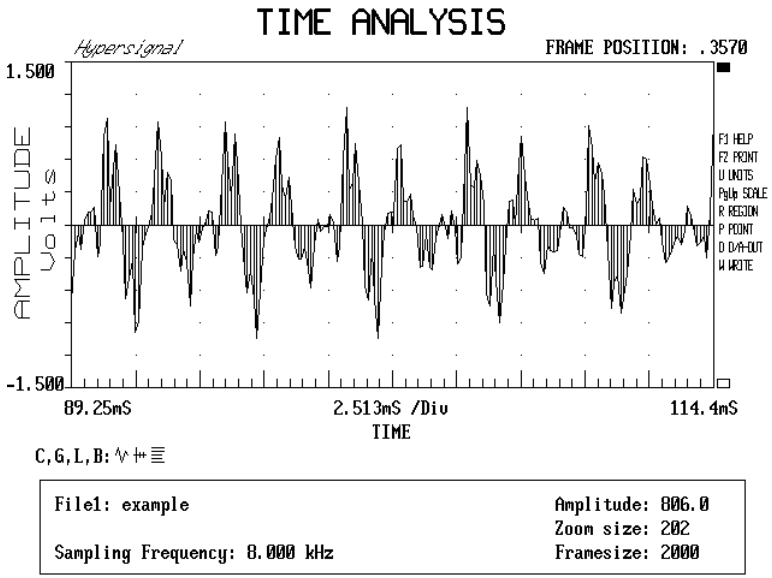


*Figure 5-5*

Note that at this level of zoom, individual samples are clearly visible. Note also that the horizontal-axis units display have been updated to show the current zoom position in the waveform.

## 5.1.1 Dual- and Overlaid-Trace Display

Now press the 'Esc' key, press it again, go to the UTILITIES column, and invoke the "User Setup" function. Set the DUAL AUTOSCALE field to 'ON', press 'Esc', and return to the Waveform Display/Edit function. Now enter the following Waveform Display/Edit menu:

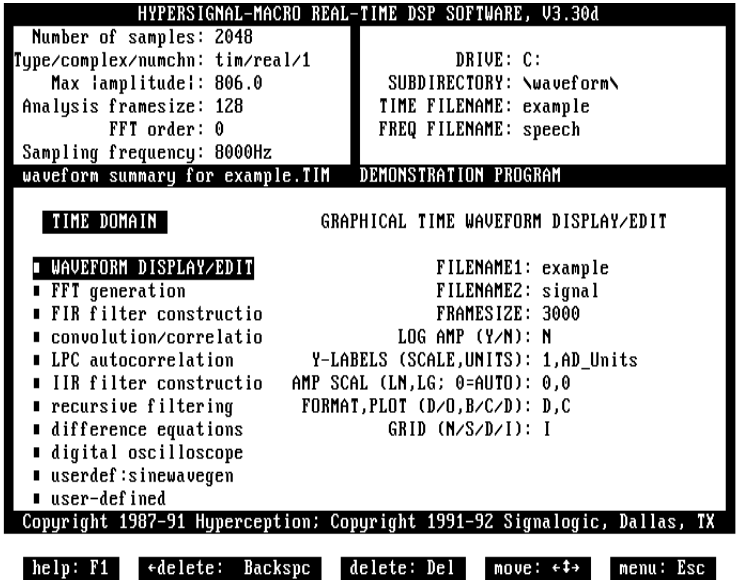


*Figure 5-6*

Because both the FILENAME1 and FILENAME2 fields are non-blank, this menu will produce a “dual” display of the example and signal waveforms. Note the FORMAT field entry of ‘D’, which specifies dual-trace display (an ‘O’ entry would specify overlaid-trace display). After waveform display occurs, you should see that the two traces are similar in terms of ampltitude.

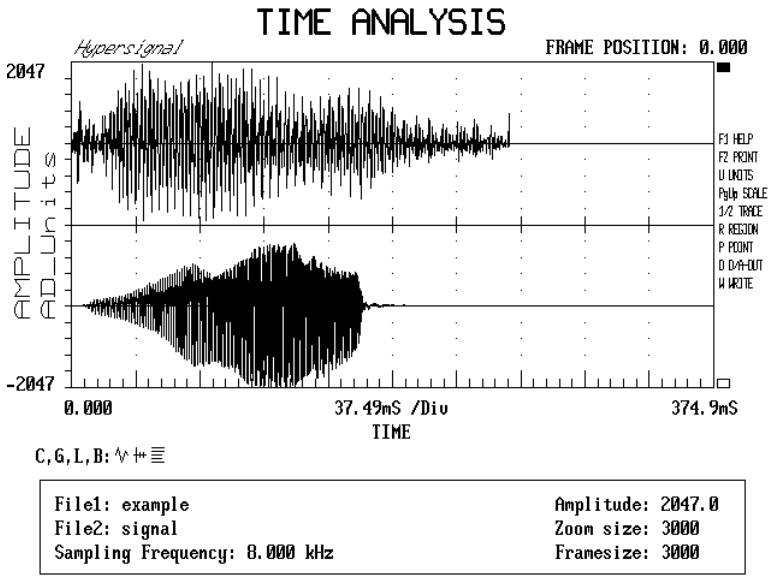


*Figure 5-7*

Now press the ‘Esc’ key, press it again, go to the UTILITIES column, invoke the “User Setup” function, and reset the DUAL AUTOSCALE field to ‘Off’. Now return to the Waveform Display/Edit function. Now you can see the actual relative difference between example and signal. The example waveform will be much smaller if it is the “speech” file resulting from File Acquisition (see section 4.1, Acquiring Data From Disk Files). What you did here was temporarily turn on “dual autoscaling”, which causes the program to scale the smaller of two waveforms to match the larger, thus fully utilizing the available display area and allowing easier comparison for data dissimilar in amplitude. The DUAL AUTOSCALE field is a global parameter; its current setting applies to all Hypersignal dual waveform displays.

Try the same display after setting the FORMAT field in the data-entry menu to ‘O’. You should see overlaid-trace display, with each trace shown in a different color. By invoking the ‘C’ (color) command during waveform display, you can change the color combinations (press the ‘Enter’ key to cause the waveform to redraw and the new color settings to take effect).

Note that in dual-trace or overlaid-trace display, the ‘1’ and ‘2’ keys can be used to select the “currently active waveform trace”, which is indicated by a little rectangle at the right of the waveform data area. The sampling frequency, units display, current editing mode, and other waveform display settings will change to match the currently selected trace as the ‘1’ and ‘2’ keys are used.

Pressing the ‘Enter’ key while the cursor is positioned at the last field will advance to the actual graphical display screen.

### 5.1.2 Time Domain Log Amplitude Display

To see time domain display similar to other examples in this chapter, but with log amplitude display, enter the Waveform Display/Edit menu shown in Figure 5-1, but with the LOG AMP field enabled (set to ‘Y’). The resulting display, as shown in Figure 5-8 below, shows data as 20log10(x), where x refers to time domain data points.

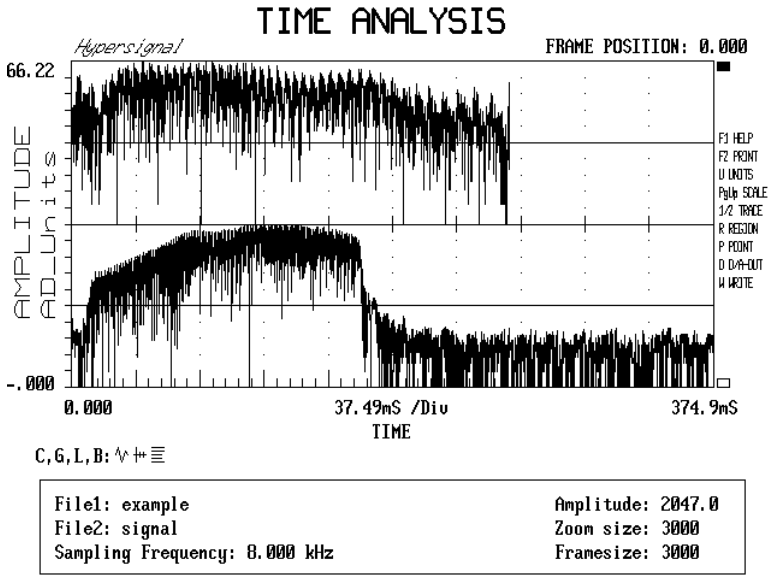


*Figure 5-8*

Functions such as zoom, editing, pan, etc. work as in linear amplitude display mode. The only function that operates differently in log mode is ver-

tical scaling: pressing the ‘PgUp’ and ‘PgDn’ keys will cause the display to expand and contract vertically, as in linear mode, but with the top axis fixed, or “pinned.” Also, the ‘S’ key can be used at any time during waveform display to toggle back and forth from linear to log display.

## 5.2 Time Domain Waveform Editing

Now press the ‘Esc’ key and enter the following Waveform Display/Edit menu:

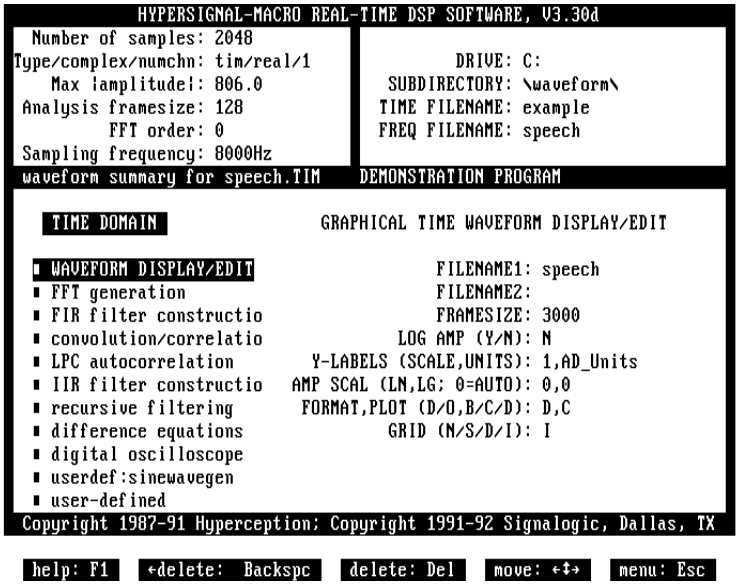


*Figure 5-9*

Once waveform display is complete, position the zoom box so that it outlines about the middle 1/3 of the waveform. Now press the ‘W’ key. You should be prompted for a new waveform filename. Enter “chunk” followed by the ‘Enter’ key. What you have done here is write the segment of the waveform outlined by the zoom box out to a new waveform file, which you can then operate on with all Hypersignal functions as with any other waveform. If the zoom box is not currently displayed, then the whole screen will be written out to the new file.

Now re-position the zoom box so that it again outlines about the middle 1/3 of the waveform. Now press the ‘Delete’ key on the keyboard. You will then be prompted to either delete (‘D’) or copy (‘C’) the zoom box area to the “paste file”, a temporary data file that holds intermediate results when doing cut-and-paste waveform editing. Press the ‘D’ key. After a few moments the waveform will re-display, with the outlined segment removed. Now position the zoom box to the right of the display area, beyond all waveform data. Now press the ‘Insert’ key on the keyboard. You will then be prompted to either insert (‘I’) or overwrite (‘O’) the current paste file contents into/over the waveform starting at the left edge of the zoom box. In this case, since you are beyond the end of the waveform, both inserting and overwriting will do the same thing. Press the ‘I’ key. After a few moments you will see the data that you previously deleted appear at the insert position. Note that the “dead” area between the old end of the waveform and the newly inserted area is filled in with zeros.

This type of editing, “cut-and-paste” editing, can be performed at any level of zoom. The samples *between and included by the edges* of the zoom window are operated upon. Thus, by greatly magnifying the data, you could delete a single sample by reducing the zoom window to “zero width” (a vertical line) and placing it directly on top of the sample to be deleted.

Now, press the ‘P’ key. This will take you from “region” mode, in which cut-and-paste editing is enabled, to “point” mode, in which “rubber-band” editing is enabled. Note that moving the mouse (or arrow keys) now controls a crosshair cursor, instead of a target window. Now press the ‘Home’ key (or middle button on the mouse). This will “tack” you to the waveform. Mouse motion (arrow keys) will now draw a new waveform shape in a different color, while still leaving the previous waveform shape. Press the ‘Home’ key (or middle button on mouse) again to “untack” from the displayed waveform. With little or no zoom in effect, you can use this mode to make gross changes in the waveform, perhaps to draw your own function, or re-shape a segment of data. At high levels of zoom you can use this mode to smooth out a glitch, or smoothly connect segments that were pasted together using the cut-and-paste editing mode.

Now, press the ‘Enter’ key to redraw the display. Note that the new rubber-band editing changes have “taken hold”, and the previous waveform shape is no longer displayed. This would happen for any operation that caused re-display: zoom, pan, switching traces, etc.

Now, press the ‘R’ key to return to region mode. Position the zoom box around part of the area that you edited in point mode. Zoom in to see how the waveform has been modified in greater detail.

**POWER NOTE S: The Waveform Display/Edit function can be used to edit any waveform, not just time waveforms. Enter the filename plus extension in the FILENAME1 and FILENAME2 fields to override the default .TIM extension.**

**When editing in dual- or overlaid-trace display, the currently active trace for editing controls is controlled by the ‘1’ and ‘2’ keys—see section 5.1.1, Dual- and Overlaid-Trace Display, above.**

## 5.3 Hardcopy Print-out

While still inside the graphical display, press the ‘F2’ function key on the keyboard. You will see a “pop-up” printer setup menu appear at the top-middle of the display. In this menu you can set various parameters that control the final image that will appear on your system’s printer. The first parameter, PRINTER NUMBER, specifies the printer manufacturer and model. Use the up and down arrows to change the current PRINTER NUMBER selection (all selections are in alphabetical order). Once you have selected the type of printer currently installed in your computer system, use the right arrow key to move to the next field, OUTPUT. Use the up arrow key to toggle through the available selections until “FILE” appears (note that you could type ‘FILE’ in directly, also). You cannot edit the filename field; this field is set automatically by the software. Now use the right arrow key to advance to the next field, XRES (horizontal dots-per-inch). Again, the up and down arrows will show the available selections. If you know the maximum value for your printer, you can enter it in directly (pressing the ‘Enter’ key will then go to the next field). We suggest 300 in this example, because even if your printer can’t go this high, it should use its highest possible value. Now move to the XLEN field, and set to ‘6’ (inches). Now move to the XSCALE field and set to ‘1’. Set the YRES field as appropriate (same as XRES for many printers), set the YLEN to ‘4’, (inches) and set the YSCALE field to ‘1’. Set the PORTRT/LNDSCP field to ‘P’, the BCKGND field to ‘W’, and the EMS

MEM REGENERATE/SCREEN DUMP field to ‘S’. You should have the following menu entered:

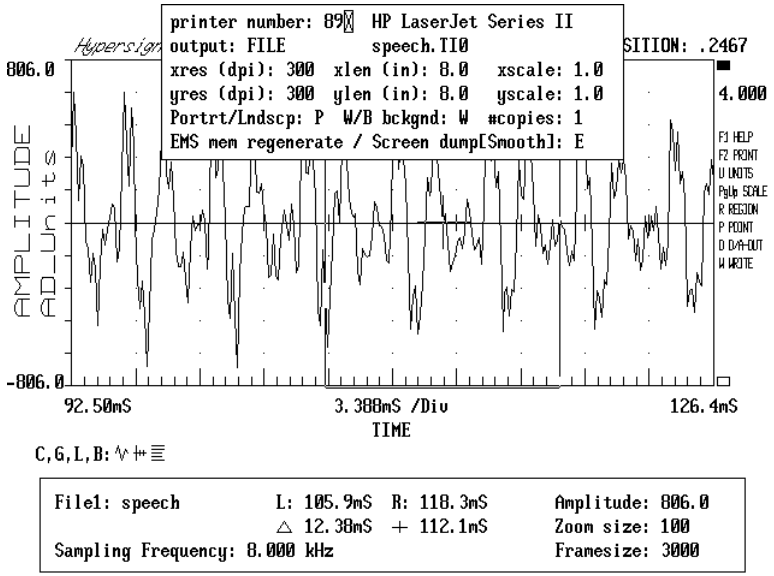


*Figure 5-10*

In the above figure, the PRINTER NUMBER field has been set to ‘89’ (HP Laser Jet Series II) for example purposes only; this field should be set for the currently installed printer in your system.

Of course, you can set the OUTPUT field directly to ‘LPT1’, ‘COM1’, etc., but for now we will simply print to a file. The XRES and YRES fields depend completely on your printer; normally, the up and down arrows will toggle through acceptable values. Usually, if a value is not exact, the printer will use whatever legal setting it allows that is closest to the value. In the above example we are using 300 in the hope that your printer will default to its highest value. Consult the documentation for your printer to learn its maximum dots-per-inch values exactly.

Portrait mode (‘P’) tells the software to print the image without any rotation, landscape mode (‘L’) causes the image to be rotated 90 degrees. A white background (‘W’) is the typical setting for black/white printers unless you desire a “reverse video” image. However, for color printers, a black background (‘B’) is suggested.

Now, we must discuss the difference between EMS memory regeneration and a straight screen dump, as well as the interaction between this field and the XLEN/YLEN and XSCALE/YSCALE fields. In the example above, we have selected a straight screen dump. This means that whatever image data is currently contained in the memory on your video card will be sent to the printer. For a screen dump, the XLEN and YLEN fields have no effect (since the image has already been created and exists in video memory) and only the XSCALE and YSCALE are active. If, for example, you increased XSCALE and YSCALE to 2 (two), then each pixel in video memory would be multiplied by 2 in the x and y directions and become a block of four pixels in the printer image. This would definitely increase the size of your image, but not its resolution: the image becomes bigger, but also "blockier". Of course, you can do the same thing by varying the XRES and YRES fields, but we are assuming the maximum dots-per-inch setting for the printer will always be used.

For an EMS memory regeneration, the entire image is redrawn in EMS memory, and then sent to the printer. This allows extremely high-quality, or "publication-quality", print-outs to be created. If EMS memory is not installed on your machine, or there is not enough of it, an error message will appear informing you of this. When doing an EMS memory draw, the XLEN and YLEN fields, as well as the XSCALE and YSCALE fields, are active. Since we are re-drawing the image, we can use the XLEN and YLEN fields to exactly control the width and height (in inches) of the image according to the current XRES and YRES dots-per-inch settings. The XSCALE and YSCALE fields still work as usual, but become somewhat redundant with the XRES and YRES fields active; since they will only serve to coarsen the image resolution, their usefulness with EMS memory regenerations is dubious, and they should be set to one.

An EMS memory print caveat: EMS memory regenerations *take time and EMS memory*. Usually such prints involve much higher resolutions than screen dumps; transferring millions of pixels to the printer can take up to 20 minutes or more for a slow system and a huge image. Likewise, you must have enough EMS memory available at the time of printing (if not, an error message will be displayed at the lower right of the waveform data area). This mean arranging the sequence in which memory resident utilties and other programs are invoked to allow enough EMS memory to be free at the time the print-out is needed.

After printing is complete, you could use the DOS "copy" command to send the file to the correct printer port. In fact, you can do this from

Hypersignal. First, press the ‘Esc’ key a couple of times to return to the main menu, then press the ‘F10’ key. This is the DOS “hot-key” that invokes a temporary DOS shell. Now enter a DOS copy command:

copy example.TI0 lpt1:

(Use the command “copy example.TI0 lpt1: /b” for DOS versions 4.0 and higher). Of course, your printer might be connected to lpt2, com1, com2, etc. so use the correct output device in the copy command. Type ‘Exit’ followed by the ‘Enter’ key to return directly to Hypersignal from the DOS shell. If you don’t want to save print-outs to intermediate files, you can enter the output device directly into the pop-up Hypersignal printer-setup menu also.

## 5.4 D/A Playback From Displays

The following section applies only to real-time software versions. First, enter the following Waveform Display/Edit menu:

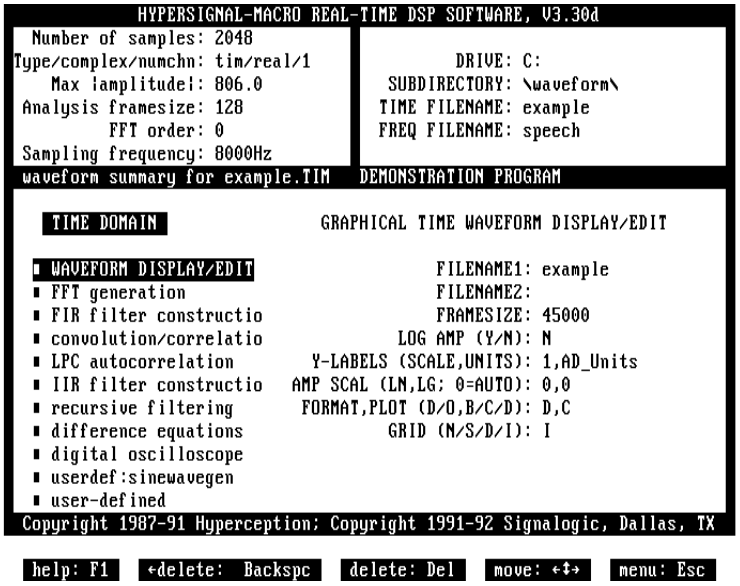


*Figure 5-11*

This should display the entire waveform that you acquired in chapter 4, ACQUIRING DATA. Once the waveform is displayed, position the zoom box so that it outlines about the middle 1/3 of the waveform. Now, press the 'D' key (D/A output). If you have your speaker still connected to output channel 0, you should hear the segment of data inside the zoom box. If no zoom box is displayed, the entire screen will be sent to the D/A. Note that you can isolate even very short segments of data for playback—an excellent example might be the consonants in a speech recording. By returning to the data entry menu and setting the FRAMESIZE field as necessary, huge waveforms (of any size) can be sent continuously to the D/A output. Use the 'Esc' key or another 'D' key to abort a D/A playback in progress.

The D/A Playback function is also available in the 2-D Spectrograph frequency domain display—see section 7.1, 2-D Spectrograph (Contour) Display.

**POWER NOTE: When performing D/A playback in dual- or overlaid-trace display, the currently active trace for playback is controlled by the '1' and '2' keys—see section 5.1.1, Dual- and Overlaid-Trace Display, above. Note that the sampling frequency used for the D/A playback is that of the waveform in the currently selected display trace.**

*this page intentionally left blank*

# 6.0 FFT GENERATION

This chapter of the tutorial discusses typical types of FFT Generation that can be performed in Hypersignal, including arbitrary time data and filter impulse responses, are covered. Overlap, windowing, and time framesize are discussed.

First we will apply an FFT to the “example” waveform we acquired in chapter 4, ACQUIRING DATA. Then we will use the FFT function on a simple filter impulse response. Lastly, we will process a speech waveform using a high degree of overlap to increase the “time resolution” of the resulting frequency waveform file. The “example” waveform in the first menu below is expected to be either the result of the File Acquisition (active in *all* Hypersignal versions) or Analog Conversion (active in real-time software versions only) functions described in chapter 4, ACQUIRING DATA. Enter the “example” waveform into the following FFT Generation menu:

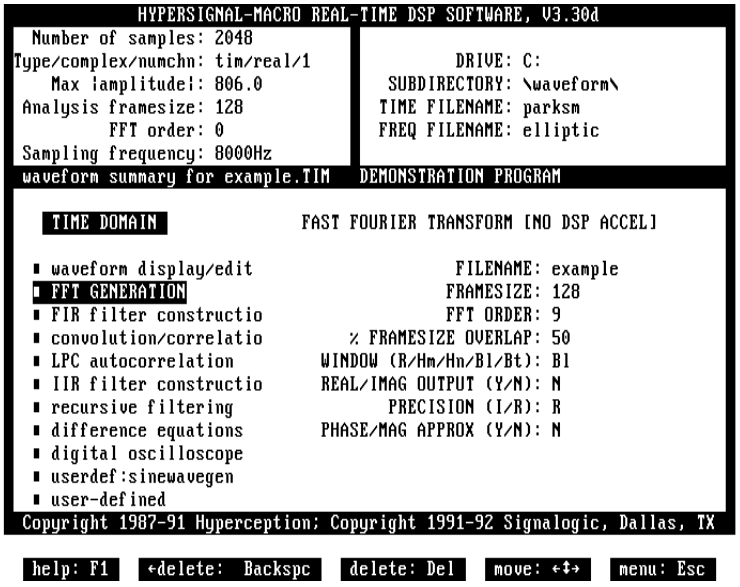


*Figure 6-1*

**NOTE: The PRECISION and PHASE/MAG APPROX fields will not appear if a DSP Accelerator board is in use. In this case, the precision is the natural wordlength of the DSP chip.**

Remember to enter the last field or press <ctrl-Enter> to cause processing to begin. In the above example, we are using a FRAMESIZE of 128 and an FFT size of 512 (FFT ORDER=9; FFT size is $2^{\text{FFT ORDER}}$). The choice of framesize here depends completely on your data and your application. Generally, FRAMESIZE should be chosen such that you feel your data remains "quasi-stationary" for this many samples (a duration of time equal to FRAMESIZE divided by the sampling frequency of the waveform). By quasi-stationary, we mean that the major frequency components in the data during that period of time didn't change suddenly; by segmenting the input waveform into successive segments of FRAMESIZE length, slowly changing frequency characteristics can be analyzed with respect to time. The 'Bl' window entry will eliminate any "edge" noise due to the segmenting of the data into arbitrary frames ('Bl' invokes a Blackman window, which tapers smoothly to zero at both ends of the FRAMESIZE length). The choice of 50% FRAMESIZE OVERLAP ensures that no noise spikes, glitches, or other small (down to single-sample duration) data features will be missed becaused they occurred at or near the frame edges, where the window value is zero or nearly zero. REAL/IMAG OUTPUT is turned off—you would only want this field enabled if you planned to work with the real and imaginary data at some later point, for example to perform an inverse FFT, or generate transfer function data.

Allow the above example to complete, then enter return to the TIME DOMAIN column and allow the following FFT Generation menu:

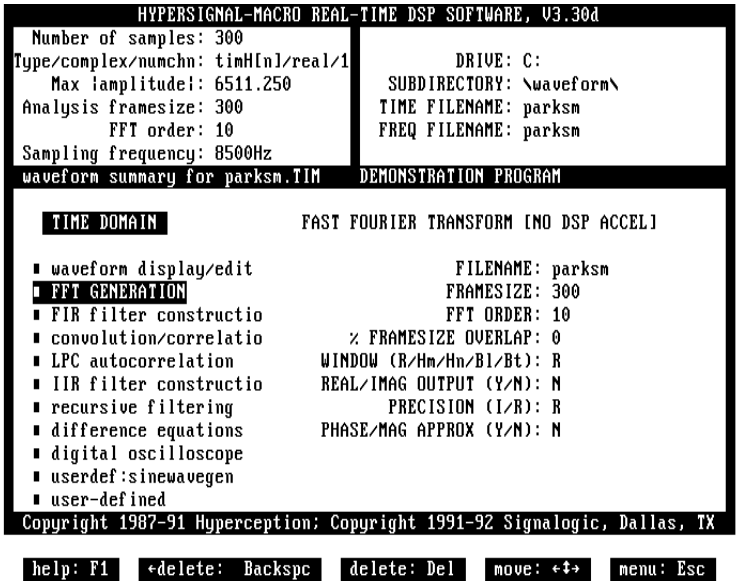


*Figure 6-2*

**NOTE: The PRECISION and PHASE/MAG APPROX fields will not appear if a DSP Accelerator board is in use. In this case, the precision is the natural wordlength of the DSP chip.**

In the above example, the framesize and FFT size are initialized to encompass the entire filter impulse response length—in this case we don't want any segmentation of data; we want to see the entire filter frequency response in one output frame, on one display. The 'R' (rectangular) WINDOW entry is alright because we have no possibility of "edge" noise due to arbitrary segmentation. Likewise, we can enter '0' for the % FRAMESIZE OVERLAP field. This time, we are turning the REAL/IMAG OUTPUT field on, because we want to use the real and imaginary output data in another example later in the tutorial.

Actually, in the above example all fields except for REAL/IMAG OUTPUT will initialize automatically once the filename has been entered because "parksm" is an impulse response waveform produced by the Hypersignal FIR or IIR Filter Construction functions.

Allow the above example to complete, then return to the TIME DOMAIN column and enter the following FFT Generation menu:

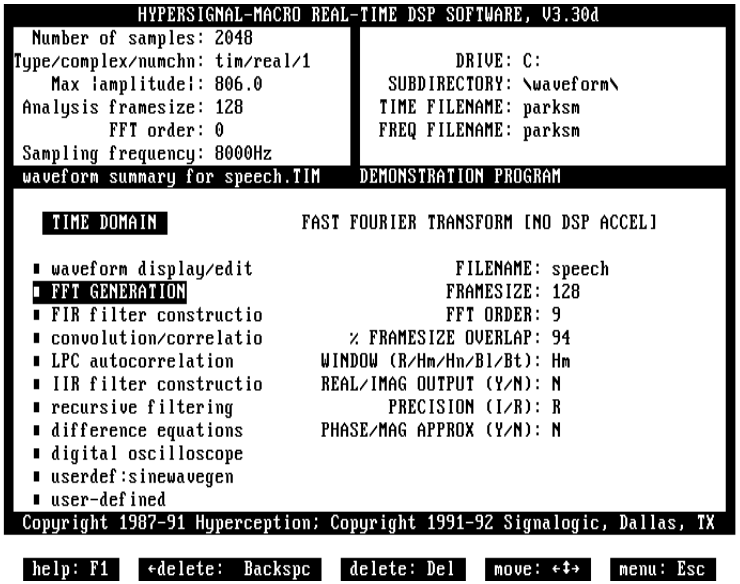


*Figure 6-3*

**NOTE: The PRECISION and PHASE/MAG APPROX fields will not appear if a DSP Accelerator board is in use. In this case, the precison is the natural wordlength of the DSP chip.**

This menu will apply a high degree of overlap to the FFT process, causing the time resolution of the resulting frequency file to be very high. The choice of 80 for FRAMESIZE means that approximately 10 msec of speech will be considered in each frame (because the speech waveform has a sampling rate of 8000 Hz). This amount of time was chosen because it is generally accepted that arbitrary segments of human speech are “quasi-stationary” for 10 to 15 msec periods—that is, the basic frequency content of the segment isn’t going to change much in that amount of time. The % FRAMESIZE OVERLAP value of 94% means that the last 75 points of any given frame will be the first 75 points of the next frame; in other words, only 5 new points will be considered in each frame. This pro-

vides very high time resolution when viewing the resulting frequency domain file in the 2-D Spectrograph or 3-D Spectrogram display functions. With this resolution, if you isolate a segment of interest in these displays, even a small segment such as a consonant in the case of the speech data used in this example, you will be able to zoom to a level of great detail, if desired.

*this page intentionally left blank*

# 7.0 DISPLAYING FREQUENCY DOMAIN WAVEFORMS

This chapter of the tutorial covers display of frequency domain waveforms using the 2-D Spectrograph (contour) display function, 3-D Spectrogram (waterfall) display function, and other types of frequency domain display. Examples include contour plot of a waveform with very high time resolution, log frequency axis display, phase unwrapping, and frame-by-frame magnitude and phase display, among others. Display from an overall perspective, display at zoom levels, panning between time segments or frames, vertical scaling, and changing units are discussed.

Before continuing we should point out that all the frequency display menus below expect waveforms that should have already been processed in the FFT Generation example in the immediately preceding chapter (chapter 6, FFT GENERATION).

## 7.1 2-D Spectrograph (Contour) Display

First, we will consider the 2-D Spectrograph function, because it may well be the most fun. It definitely allows display of the most information on one screen at one time of any frequency domain display in Hypersignal. In a 2-D Spectrograph display, multiple FFT output frames are displayed at once, with frequency along the vertical axis, time along the horizontal, and amplitude encoded either with different colors or gray shades. In this format, you are “looking down” at the frequency domain data—for this reason, the 2-D Spectrograph is often referred to as a form of “contour plot.” Enter the following 2-D Spectrograph menu:

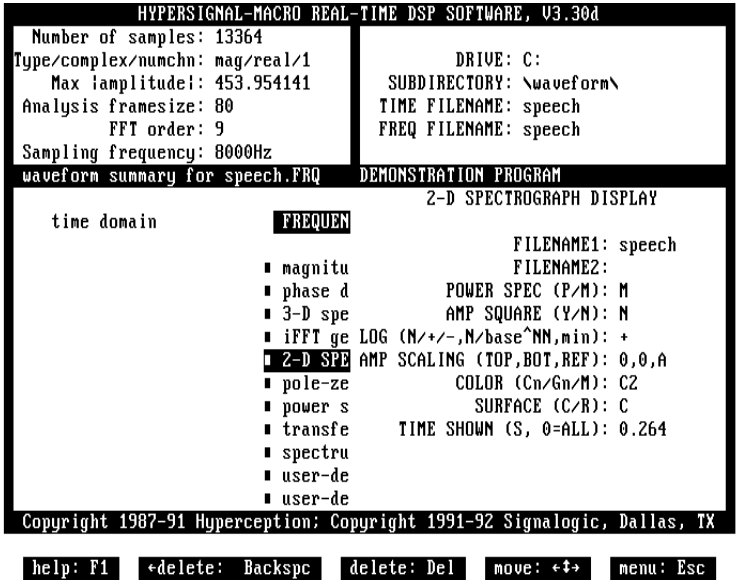


*Figure 7-1*

Remember to enter the last field or press <ctrl-Enter> to cause graphical display to begin. In the above menu, the POWER SPEC field is set to 'M' (magnitude), the AMPLITUDE SQUARE field is disabled ('N'), and the LOG field is set to enable log display for amplitude only (for example, an entry of '+,10^3,200' would enable log display—3 cycles of base 10, with minimum lefthand display of 200 Hz—for the frequency axis also). The AMP SCALING field has been set to '0,0,A' to specify autoscaling, with an "amplitude step" of 5dB per color.

**POWER NOTE:** **Hypersignal divides the amplitude shown on the display into 8 equal levels; the LOG and AMP SCALING fields can be used to control the amount of amplitude and type of data spread over these levels. When using the default entry of '0,0,A', a step value of 5 dB per quantization level is preset (i.e. the same display could be achieved with an entry of '0,-40,A'). The REF subfield is used to determine the 0 dB point for log amplitude displays; possibilities include: the maximum peak in the waveform ('A' entry), an outside system reference value ('S' entry), the system reference value plus a system calibration offset ('C' entry), and the system reference value plus a stored waveform header calibration offset ('W' entry). See Users Manual sections 4.5.1, 2-D Spectrograph Display, and 4.5.3.1, Linear vs. Log Frequency Domain Display, for greater detail on the AMP SCALING field and the REF subfield in particular.**

The COLOR and SURFACE fields are interrelated. Entry of 'C1' and 'C2' specify different color schemes; it is expected that user preference will vary (also, you can define your own scheme—see the "CONFIG.DOC" file for more information on this subject). A COLOR field entry of 'G1' specifies gray shade amplitude encoding (and will only work on some VGA graphics systems—see Users Manual section 4.5.1, 2-D Spectrograph Display). A SURFACE field entry of 'C' specifies a contour plot, in which we look straight down on the data; a SURFACE field entry of 'R' (raster) turns each FFT frame on its side, using some of the available display area to indicate amplitude without the need for color or gray scale. A raster selection is intended mainly for monochromatic displays, when color or gray scale is not possible. The TIME SHOWN field controls the amount of time that will be displayed along the horizontal axis in the default display. Note that we are only displaying .26 seconds of data along the time axis. If you are using real-time software and acquired the "example" waveform from your analog signal source (see chapter 4, ACQUIRING DATA), then you may want to set TIME SHOWN to 3 seconds, to see the entire waveform. Of course, you can zoom in to show smaller amounts of time using the mouse or arrow keys once inside the graphical display.

While the cursor is on the last field in the menu, press the 'Enter' key (or any button on the mouse) to advance to the actual 2-D Spectrograph display.

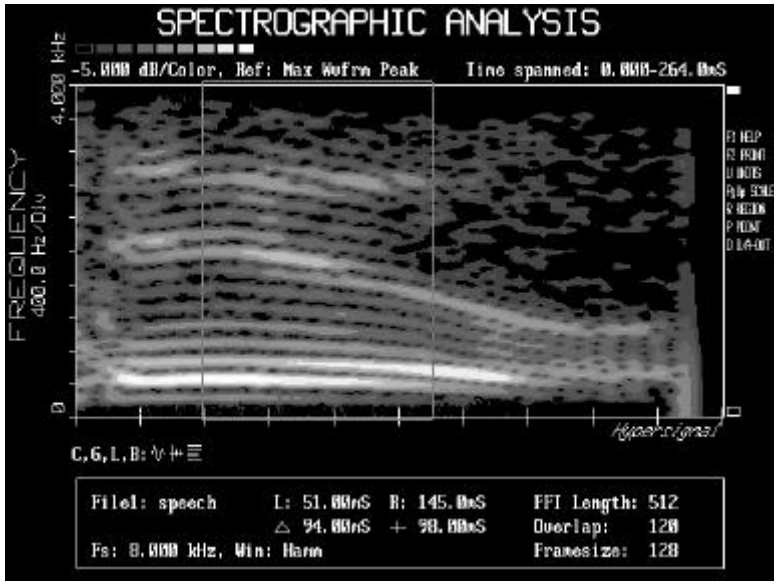


*Figure 7-2*

You should be able to clearly see the individual “formants” (3 or 4 major areas of energy) that are characteristic of human speech. Remember that frequency runs along the vertical axis, so a formant slowly moves up or down as its frequency changes. Now move the mouse around (or use the arrow keys on the keyboard) until the target window (zoom box) outlines about the middle 1/3 of the display. Right and left will translate the box; up and down will expand and contract the box. As you move the zoom box around the left and right edge positions and the “delta” between them is continuously updated at the bottom of the display. Note that you can press the ‘U’ key to change the current units of this readout.

Now press the middle key on the mouse (or ‘Home’ key on the keyboard) to zoom to a higher level of magnification.

After zooming, press the space bar key (or righthand button on the mouse) to pan to the next screen of data, and the ‘Backspace’ key (or lefthand button on the mouse) to pan to the previous screen of data.

You can use the outer two buttons on the mouse to back up from a zoom level to the previous level (16 levels of “zoom history” are available).

If you are using real-time software, you can press the ‘D’ key to send the segment of waveform outlined by the zoom box to the D/A output on your currently installed DSP/Analog board. This function operates similarly to its counterpart in the Waveform Display/Edit function. See section 5.4, D/A Playback from Displays.

## 7.2 3-D Spectrogram (Waterfall) Display

Next, we will consider the 3-D Spectrogram function. In a 3-D Spectrogram display, multiple FFT output frames are displayed at once, with frequency along the horizontal axis, frequency along the vertical, and time along a third, “apparent” axis created by drawing each FFT output frame slightly behind the previous frame. Enter the following 3-D Spectrogram menu:

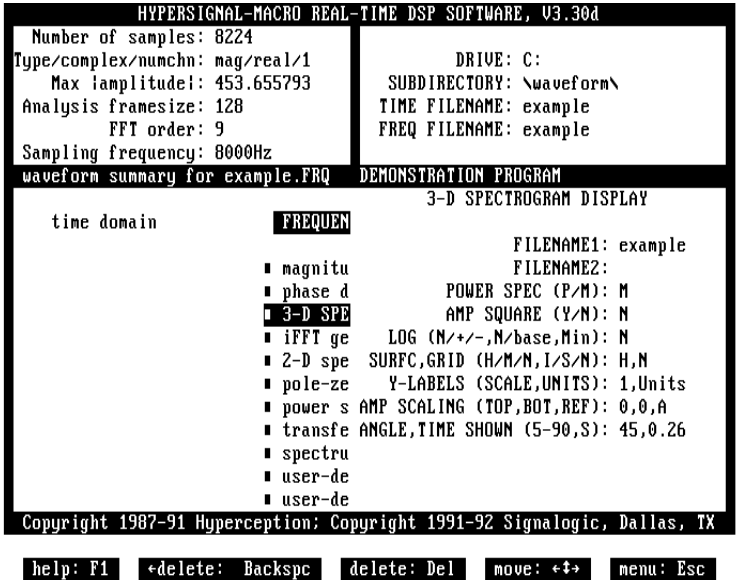


*Figure 7-3*

In the above menu, the POWER SPEC field is set to ‘M’ (magnitude), the AMPLITUDE SQUARE field is disabled (‘N’), and the LOG field is disabled for both amplitude and frequency axes. The SURFACE,GRID field is set to ‘H’ to cause hidden-line removal (‘M’ would add a “texture” mesh; ‘N’ would disable completely) and ‘N’ for no grid line display (‘I’

would specify grid line “intersection point” display). The Y-LABELS field is set for no scaling (scale factor=1) and a label of “ADus” (A/D units) to be printed along the vertical axis. The AMP SCALING field is set to ‘0,0,A’ to force the program to use the maximum value found in the waveform as the maximum vertical axis value (autoscaling). The ANGLE field controls the angle of the “apparent axis” mentioned above; 45 degrees is basically an “isometric” view, 90 degrees would be a “waterfall display”, in which each successive FFT output frame is directly behind the previous frame. The TIME SHOWN field controls the amount of data displayed in each default screen, and should be specified in seconds.

**POWER NOTE:** **The REF subfield is used to determine the 0 dB point for log amplitude displays; possibilities include: the maximum peak in the waveform (‘A’ entry), an outside system reference value (‘S’ entry), the system reference value plus a system calibration offset (‘C’ entry), and the system reference value plus a stored waveform header calibration offset (‘W’ entry). See Users Manual sections 4.5.1, 2-D Spectrograph Display, and 4.5.3.1, Linear vs. Log Frequency Domain Display, for greater detail on the AMP SCALING field and the REF subfield in particular.**

Pressing the ‘Enter’ key (or any button on the mouse) while the cursor is positioned on the last field will advance to the actual graphical display screen.

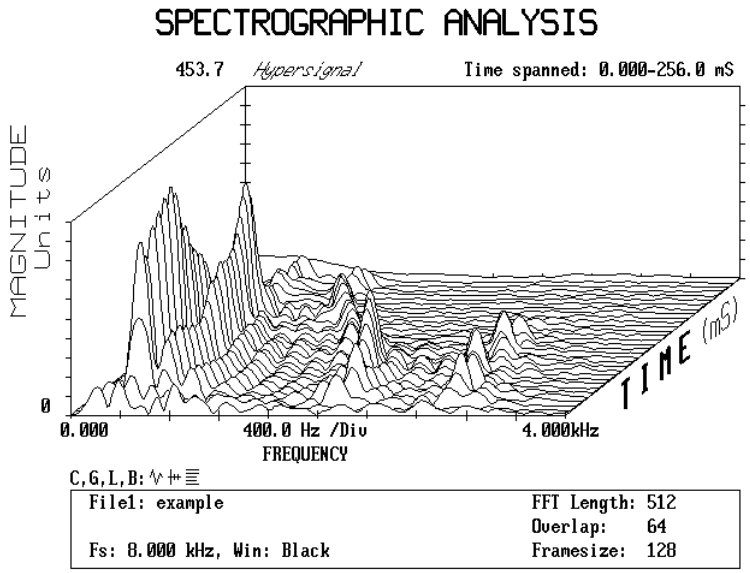


*Figure 7-4*

Note that we are only displaying .26 seconds of data in the time "dimension". If you are using real-time software, and acquired the "example" waveform from your analog signal source (see chapter 4, ACQUIRING DATA), then you may want to set the TIME SHOWN field to 3 seconds, to see the entire waveform. By changing the LOG field entry to 'N,3^10,100', which specifies a log10 frequency axis with a minimum display value of 100 Hz, the following display should result:

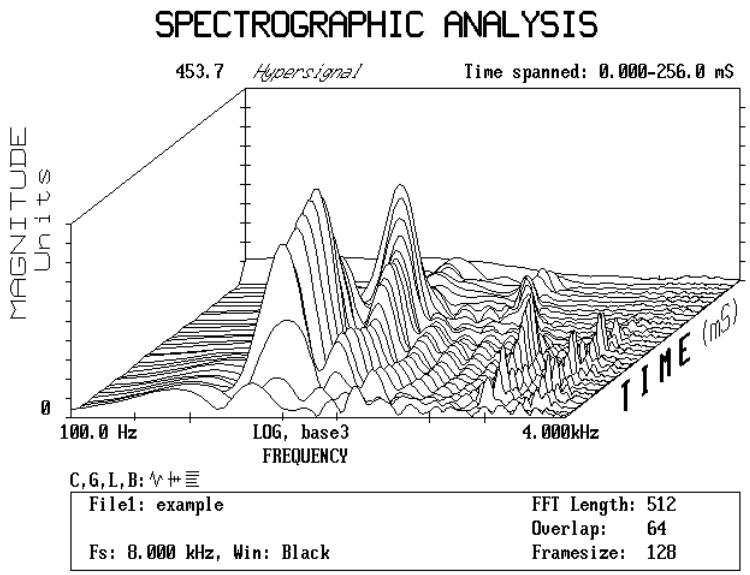


*Figure 7-5*

Use the arrow keys (or move the mouse) until the target window ("zoom box") is positioned so that it occupies about the middle 1/3 of the display area. The left and right arrow keys (or mouse motion) move the zoom box from side to side; the up and down arrow keys (or mouse motion) expand and contract the size of the zoom box.

Once positioned, press the 'Home' key (or middle button on the mouse) to zoom in on the waveform. The waveform area contained in the zoom box at this time will become the new display.

Repeat the zoom process a few times until it feels comfortable. Note that as you magnify the data, individual samples become visible.

To back up to previous levels of zoom, use the outer two buttons on the mouse. A maximum of 16 levels of "zoom history" are stored by the program.

To pan to the next segment of data (in the time dimension, or along the "apparent" axis), use the space bar key (or righthand button on the mouse). To pan to the previous segment of data, use the 'Backspace' key (or lefthand button on the mouse). Note that the pan function is always active, regardless of the current amount of time shown. Each segment represents

an amount of data specified by the TIME SHOWN field; use the “Time Spanned” printout at the upper right of the display to locate your current position within the frequency domain waveform.

Try pressing the ‘U’ key a few times; this should toggle the current frequency units from their current SI value to 10e3 higher, then to 10e-3 lower, then to “point” readout (in FFT “bins”), then back to the default setting.

To expand and contract data vertically, press the ‘PgUp’ and ‘PgDn’ keys, respectively.

## 7.3 Magnitude Display

Next, we will consider the Magnitude and Phase Display functions. These are “x-y” displays that show a single frame of magnitude or phase at one time, and allow cursor read-out and measurement on individual samples (which are really output FFT “bins”). First, enter the following Magnitude Display menu:

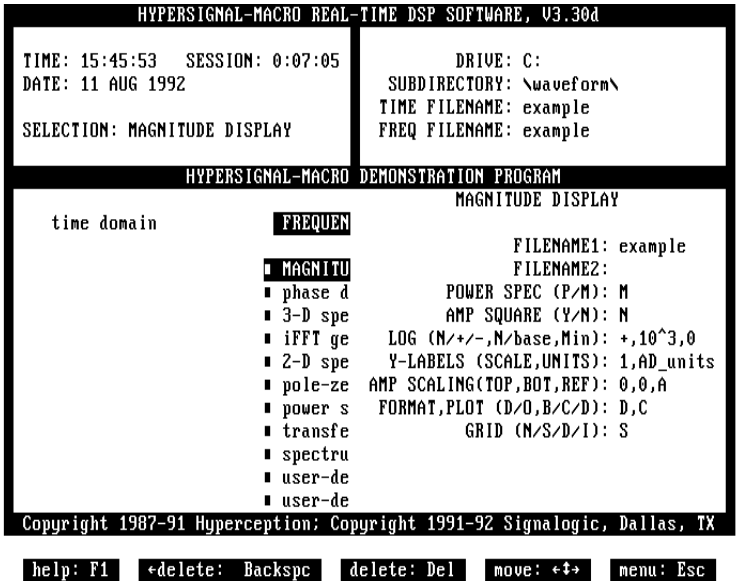


*Figure 7-6*

This menu will cause the program to utilize the full display area to show

the first output FFT frame of the "example" frequency waveform. The POWER SPEC field has been set to 'M' for magnitude display, and AMP SQUARE has been set to 'N' to disable amplitude-squared display. The LOG field entry of '+,10^3,0' specifies log format on the vertical axis (the '+') and also 3 cycles of base 10 log display on the horizontal (frequency) axis. Because we have specified log amplitude display, the Y-LABELS field is inactive. However, if linear display had been specified the Y-LABELS field entry would cause no scaling to occur (scale factor=1), and a label of "AD_units" to be printed on the vertical axis. The AMP SCALING field is set to '0,0,A' to force the program to use the maximum value found in the waveform as the "0 dB" (zero dB) value. A FORMAT entry of 'D' specifies dual-trace display (ignored in this case, since the FILENAME2 field was left blank). A PLOT entry of 'C' asks for "connected" waveform depiction, where individual samples are connected by a straight line (a 'D' entry would show "discrete" sample values, and a 'B' entry would show both). A GRID entry of 'I' asks for "intersected" grid display, in which a dot appears wherever tic marks on the horizontal and vertical axes would intersect ('S' indicates solid grid lines, 'D' indicates dashed lines, and 'N' disables grid display entirely).

**POWER NOTE:** **The REF subfield is used to determine the 0 dB point for log amplitude displays; possibilities include: the maximum peak in the waveform ('A' entry), an outside system reference value ('S' entry), the system reference value plus a system calibration offset ('C' entry), and the system reference value plus a stored waveform header calibration offset ('W' entry). See Users Manual sections 4.5.1, 2-D Spectrograph Display, and 4.5.3.1, Linear vs. Log Frequency Domain Display, for greater detail on the AMP SCALING field and the REF subfield in particular.**

Pressing the 'Enter' key (or any button on the mouse) while the cursor is positioned on the last field will advance to the actual graphical display screen.

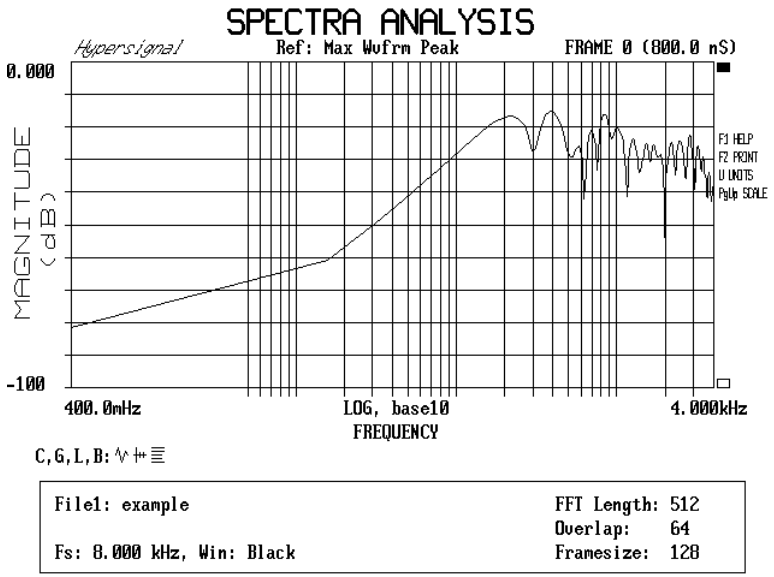


*Figure 7-7*

Now use the arrow keys (or move the mouse) until the target window ("zoom box") is positioned so that it occupies about the middle 1/3 of the display area. The left and right arrow keys (or mouse motion) move the zoom box from side to side; the up and down arrow keys (or mouse motion) expand and contract the size of the zoom box.

Once positioned, press the 'Home' key (or middle button on the mouse) to zoom in on the waveform. The waveform area contained in the zoom box at this time will become the new display.

Repeat the zoom process a few times until it feels comfortable. Note that as you magnify the data, individual samples become visible.

To back up to previous levels of zoom, use the outer two buttons on the mouse. A maximum of 16 levels of "zoom history" are stored by the program.

To pan to the next frame of data, use the space bar key (or righthand button on the mouse). To pan to the previous frame of data, use the 'Backspace' key (or lefthand button on the mouse). Note that the pan function is always active, regardless of the current level of zoom. Also note that for any level of zoom below the initial default, the pan function causes the display

to move forwards/backwards by the output FFT framesize, as opposed to the current zoom framesize as in done in time domain display. This means that some points may be skipped; but the idea is that a feature in the data worth zooming on probably has changed frequency position very little in the previous or next frames.

Try pressing the ‘U’ key a few times; this should toggle the current time units from their current SI value to 10e3 higher, then to 10e-3 lower, then to “point” readout (in samples, or FFT “bins”), then back to the default setting.

To expand and contract data vertically, press the ‘PgUp’ and ‘PgDn’ keys, respectively.

Now, press the ‘Esc’ key (or the left two buttons together on the mouse) and enter the following Magnitude Display menu:

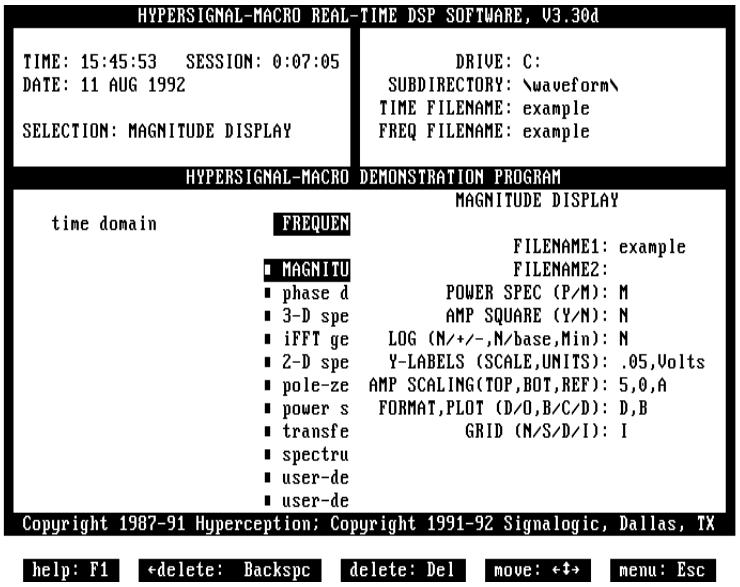


*Figure 7-8*

This menu is identical to our first one except that we have changed the vertical data format to linear (instead of log) and specified some real-world units, and we have changed the data depiction style. The scale factor value comes from dividing the full-scale voltage of the source of the input data by the resolution of the source. For example, if an analog board was used

that had 5 Volts full-scale and 16-bits of input, this would be 5/32767, or .1526e-3. For data obtained with 3 V full-scale and 12-bits resolution, the factor would be 3/2047, or 1.465e-3. In the above example we have accordingly changed the label that will be printed on the vertical display axis from “AD_units” to “Volts”. We have also changed the AMP SCALING “TOP” value to be 5, since we are now working in Volts and don’t expect any data higher than 5 Volts. Leaving the TOP subfield set to ‘0’ would still tell the program to set the maximum vertical axis value to the maximum magnitude found in the waveform. The last changes we made were to the PLOT field, which is now set for both “connected” and “discrete” data depiction, to better see individual samples, and to the GRID field, which is now set to dashed-line mode.

Pressing the ‘Enter’ key (or any button on the mouse) while the cursor is positioned on the last field will advance to the actual graphical display screen.

Do a few zooms so that you can see the “discrete” style of sample depiction clearly.

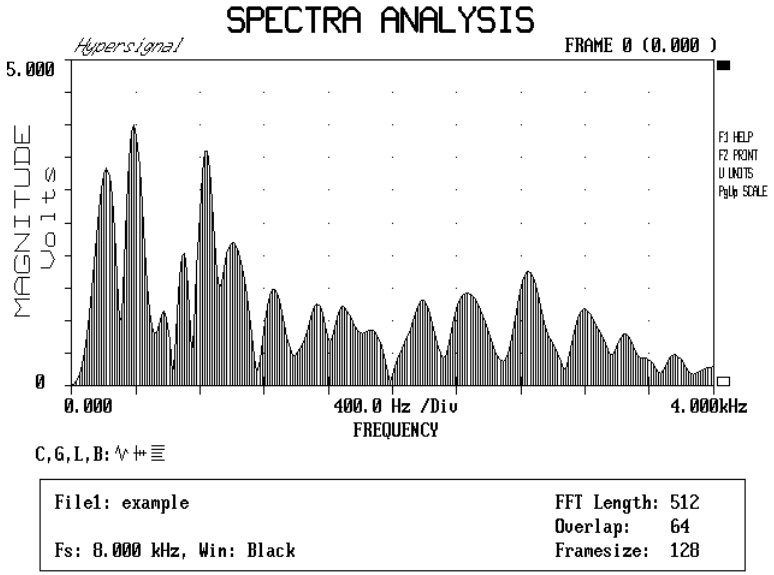


*Figure 7-9*

## 7.4 Phase Unwrapping

Now, escape to the main menu and select the Phase Display function, and then enter the following Phase Display menu:

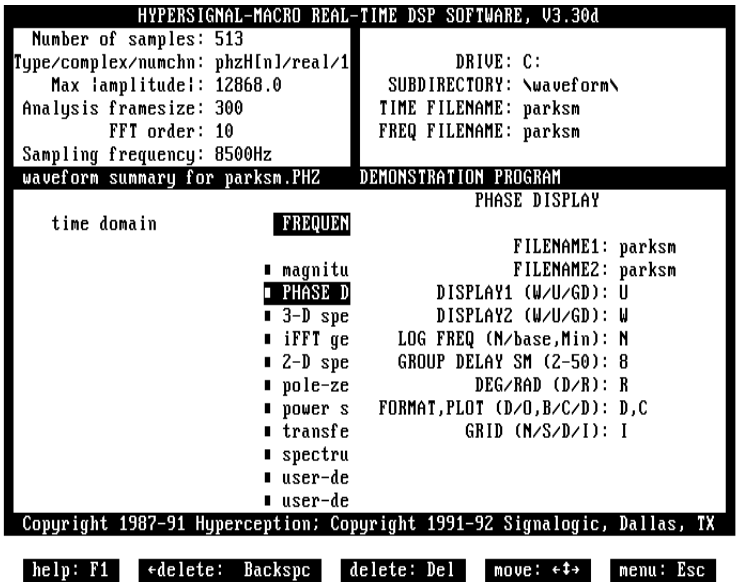


*Figure 7-10*

This menu will display the “unwrapped” phase of the parksm filter response waveform on the top trace, and the “wrapped” phase of the parksm response on the bottom trace. Phase unwrapping refers to the process of part mathematics, part educated guesswork that goes into creating a continuous phase function using data produced by the FFT function. Since the wrapped phase comes from a basic 2-quadrant ARCTAN(imag/real) operation on FFT output data, it becomes desirable to see how the phase of the data *progresses with frequency*, which requires that phase values be tracked as they go past pi and suddenly become -pi, for example. This can become extremely difficult if the data is noisy, with small magnitudes that cause the phase to “jitter” around a possible wrap-point.

A LOG field entry of ‘N’ disables frequency axis log display, the GROUP DELAY SM (group delay smoothing) field entry is ignored here because we are not asking for group delay display yet, the DEG/RAD field is set to ‘R’ for radians, a FORMAT entry of ‘D’ specifies dual-trace display (‘O’

would specify overlaid-trace display), the PLOT type is set to connected, and the GRID type is set to 'I' for intersection points.

Pressing the 'Enter' key (or any button on the mouse) while the cursor is positioned on the last field will advance to the actual graphical display screen.

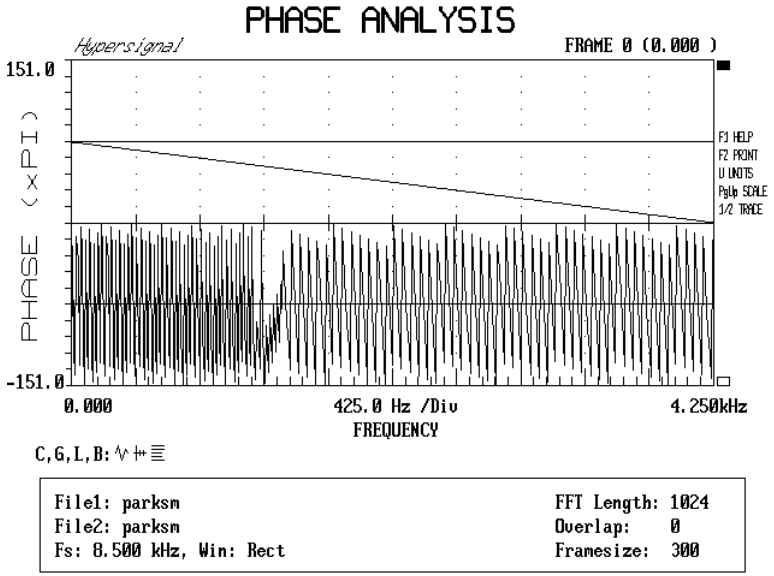


*Figure 7-11*

Note that the bottom of the screen shows what appears to be a series of jagged lines. Actually this is a continuous phase function that goes downhill but "wraps around" to higher values every so often. The top trace shows the smoothly unwrapped phase function. In this example, we are working with the phase response for an FIR filter—this is why the phase is exactly linear. Now, enter the same menu as in Figure 7-10, except change the DISPLAY1 field to a 'GD' entry. Now we are asking for group delay display on the top trace. Group delay is the negative derivative of the unwrapped phase—this is where the GROUP DELAY SMOOTHING factor comes in: to specify the number of points to use at any one time in estimating instantaneous slope of the unwrapped phase. Using a number of points provides a smoothing effect to the final group delay function, reducing error caused by phase "jitter". See Users Manual section 4.5.4.1,

Suggestions for Unwrapped Phase Display, for more information about the process of unwrapped phase display in Hypersignal.

Note that in the display the top trace is a constant, flat line. Again, this is because of the inherent linear phase of the parksm FIR filter, which delays all frequencies equally as they pass through the filter. This is not the case for IIR filters, as we shall see below. Also note that the value of group delay (displayed in msec) is exactly equal to half the FIR filter's impulse response length (use the Waveform Display/Edit function if you want to verify this). A very roundabout way of calculating an FIR filter's impulse response length!

Now enter the following Phase Display menu:

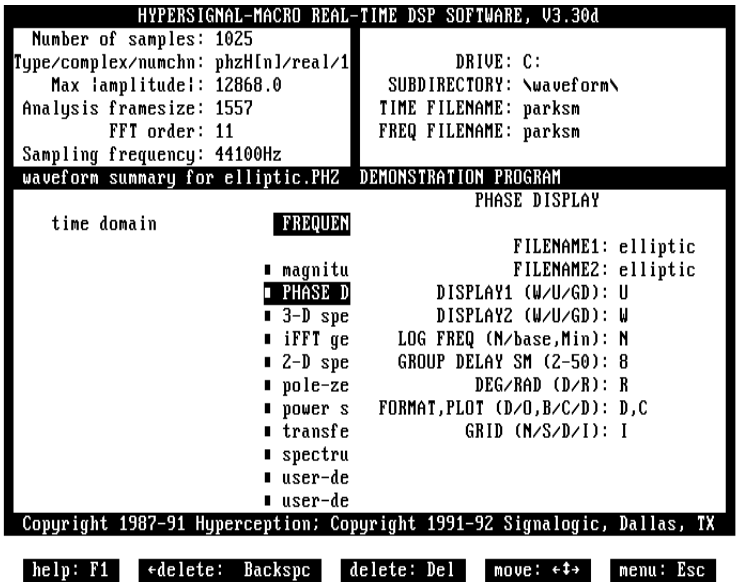


*Figure 7-12*

You should see results similar to figure 7-13, below. Note that the continuous phase of the elliptic IIR filter is definitely not linear, especially in its "transition" regions between passband and stopband. This can be one of the drawbacks to IIR filters, depending on the application, since some frequencies will be delayed more than others (frequency "smearing").

Now use the arrow keys (or move the mouse) until the target window ("zoom box") is positioned so that it occupies about the middle 1/3 of the

display area. The left and right arrow keys (or mouse motion) move the zoom box from side to side; the up and down arrow keys (or mouse motion) expand and contract the size of the zoom box.

Once positioned, press the 'Home' key (or middle button on the mouse) to zoom in on the waveform. The waveform area contained in the zoom box at this time will become the new display.

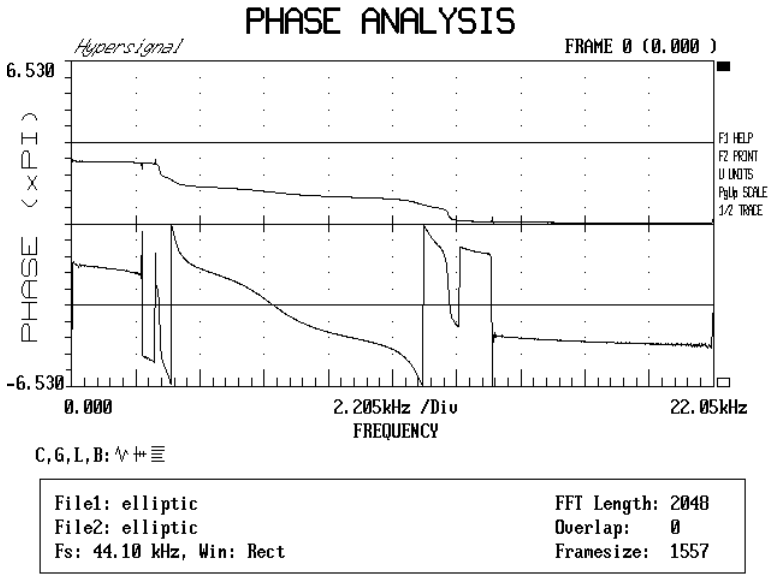


*Figure 7-13*

Repeat the zoom process a few times until it feels comfortable. Note that as you magnify the data, individual samples become visible.

To back up to previous levels of zoom, use the outer buttons on the mouse. A maximum of 16 levels of "zoom history" are stored by the program.

To pan to the next frame of data, use the space bar key (or righthand button on the mouse). To pan to the previous frame of data, use the 'Backspace' key (or lefthand button on the mouse). Note that the pan function is always active, regardless of the current level of zoom. Also note that for any level of zoom below the initial default, the pan function causes the display to move forwards/backwards by the output FFT framesize, as opposed to the current zoom framesize as in done in time domain display. This means that some points may be skipped; but the idea is that a feature in the data

worth zooming on probably has changed position in frequency very little in the previous or next frames.

Try pressing the ‘U’ key a few times; this should toggle the current time units from their current SI value to 10e3 higher, then to 10e-3 lower, then to “point” readout (in samples, or FFT “bins”), then back to the default setting.

To expand and contract data vertically, press the ‘PgUp’ and ‘PgDn’ keys, respectively.

*this page intentionally left blank*

# 8.0 DIFFERENCE EQUATIONS

This chapter of the tutorial takes you step-by-step through writing and evaluating several difference equations, both to generate new functions from scratch and to perform signal arithmetic on existing waveforms. Examples include sinewave generation, random noise generation, superposition, and heterodyning. Although the idea of "writing an equation" may seem daunting at first, you'll probably be surprised at the ease at which Difference Equations functions can be used. It certainly is an extremely flexible concept, allowing many different DSP- and math-oriented operations to be performed using a single, standardized interface. Notwithstanding these advantages, the Difference Equations function reaches its true potential when used in combination with other Hypersignal functions. For example, when combined with the D/A output function (analog signal generation, in real-time software only), Hypersignal can serve as an arbitrary function generator. Such capability might ordinarily be purchased as a stand-alone instrument, for much greater cost than Hypersignal software plus a plug-in DSP/Analog board.

**NOTE: Before continuing with the examples in this section, it should be noted that there are many additional, and generally more useful, examples of difference equations in Users Manual section 4.6.1, Writing Difference Equations.**

## 8.1 Waveform Generation

First, let's use the Difference Equations function to generate a few waveforms from scratch. As a first example, enter the following menu to design a simple sine wave:

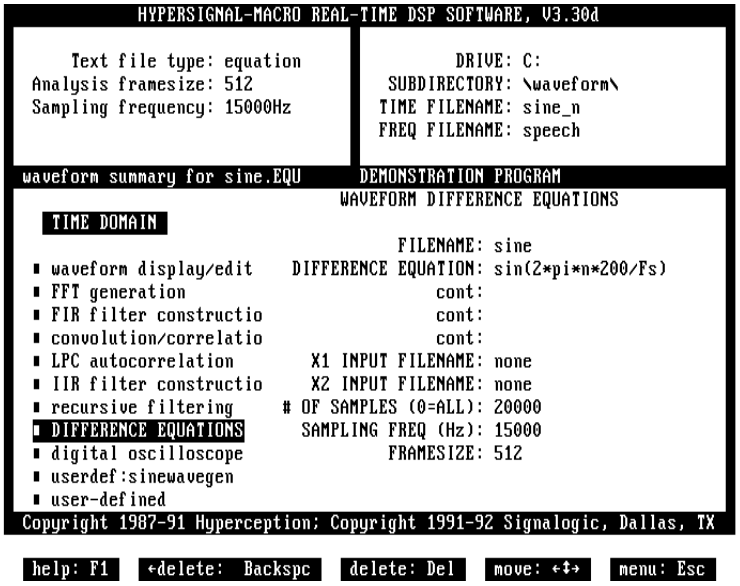


*Figure 8-1*

Okay, what about the stuff entered here? Well, the equation is specifying a 200 Hz sine wave with a maximum amplitude of 1 and a sampling frequency of 15000 Hz—note that we can refer to the value of SAMPLING FREQ we have entered in the menu by the symbol “Fs” in the equation. Note also that you can break the equation up in any way on the four lines available, because all four lines are concatenated together at run-time.

**POWER NOTES: A “y[n]=” is assumed at the beginning of the equation; this is always true unless you enter “y[ something-something ]=” at the beginning of the equation, in which case you can control the output index.**

**The “pi” in the above equation is a constant and refers to the value 3.1415... whenever used.**

**See the on-line HELP for a complete list of constants, operators, functions, and special waveform types (such as unit-step, delta, etc.) that are available in the Difference Equations function.**

Remember to enter the last field or press <ctrl-Enter> to cause processing of the difference equation to occur. Afterwards, enter a Waveform Display/Edit menu with the FILENAME1 field set to “sine”, the FILENAME2 field blank, and the FRAMESIZE field set to 512 so as to take a quick look at the resulting sine waveform:

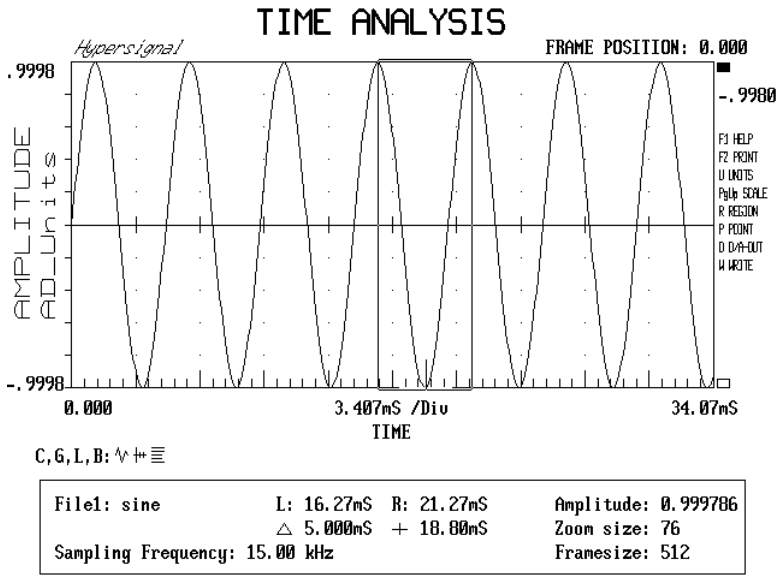


*Figure 8-2*

In Figure 8-2 above, the target window is positioned to show the period of the sine wave, which is 5 msec (1/200 Hz). Remember that you can use the arrow keys (and mouse motion) to position the target window as needed in the waveform, and then the ‘Home’ key (or middle button on the mouse) to zoom in on the displayed waveform.

As a second waveform generation example, enter the following Difference Equation menu to design a sawtooth waveform:

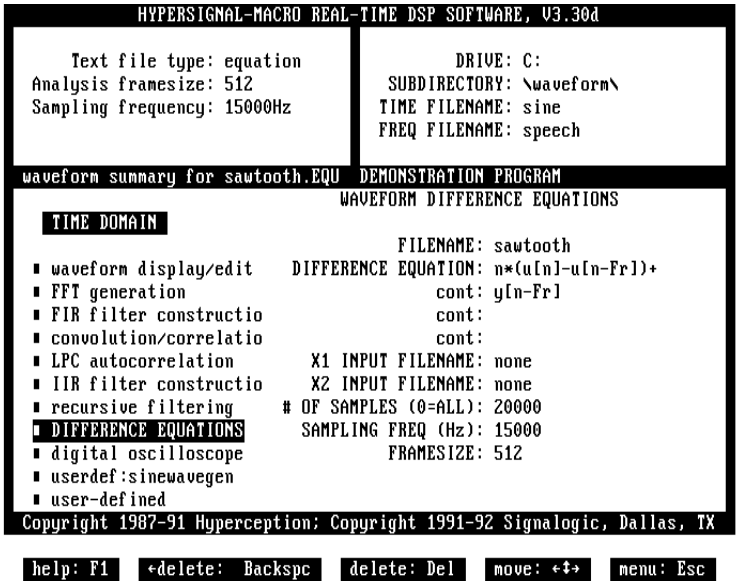


*Figure 8-3*

In this example, the maximum amplitude remains 1, and the period of the sawtooth is 512 samples. This is because we have used “Fr” to refer to the entered value of ANALYSIS FRAMESIZE for the parts of the equation that control the period of the waveform. The general form of the equation we have used can be expressed as:

$$y[n]=F(n)*(u[n]-u[n-M]) + y[n-M]$$

where F(n) is the function of the waveform described for one period, M is the length of the period in samples, u[xxx] refers to the the unit-step function, and y[xxx] refers to previous output (“recursive input”). In the above example we simply used “n” as the function F(n). This gives us a straight line with slope=1; we can then use previous output to replicate the period for a total of 20000 samples. This is possible because the use of “on” and “off” unit-step functions prevents any further *new* input after M samples. The resulting waveform would look like the Waveform Display/Edit display function (with the FRAMESIZE field set to 2000) picture shown in figure 8-4, below:

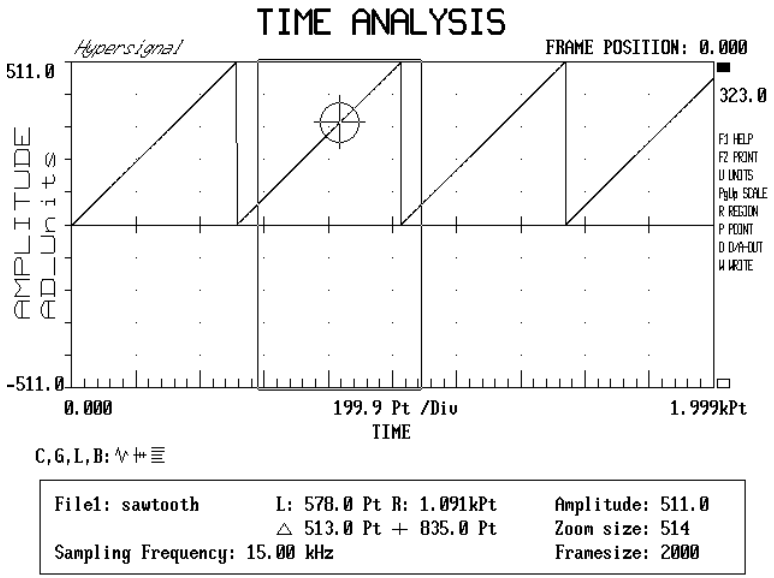


*Figure 8-4*

In Figure 8-4 above, the target window has again been positioned to indicate the period of the sawtooth waveform. Note that the units of the display have been changed to individual sample "points" (use the 'U' key). To design the sine waveform in the first example using the technique of the second example, but with an amplitude of 10000, one would use an F(n) of:

F(n)=10000∗sin(2∗pi∗n/M)

where M would be 100 (20000 Hz sampling frequency divided by 200 Hz fundamental frequency equals 100-sample period). The Difference Equations menu for this F(n) would look like:

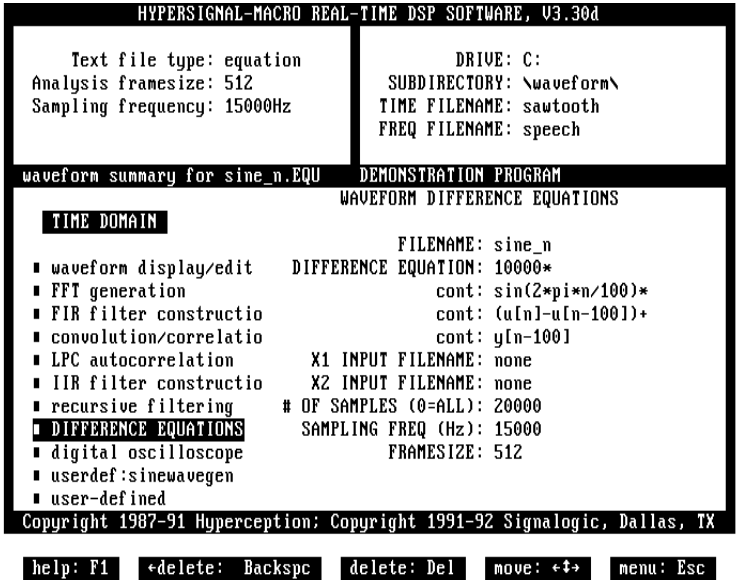


*Figure 8-5*

As a third waveform generation example, enter the following Difference Equations menu to create a sine waveform similar to the first waveform, but with some random noise added in:

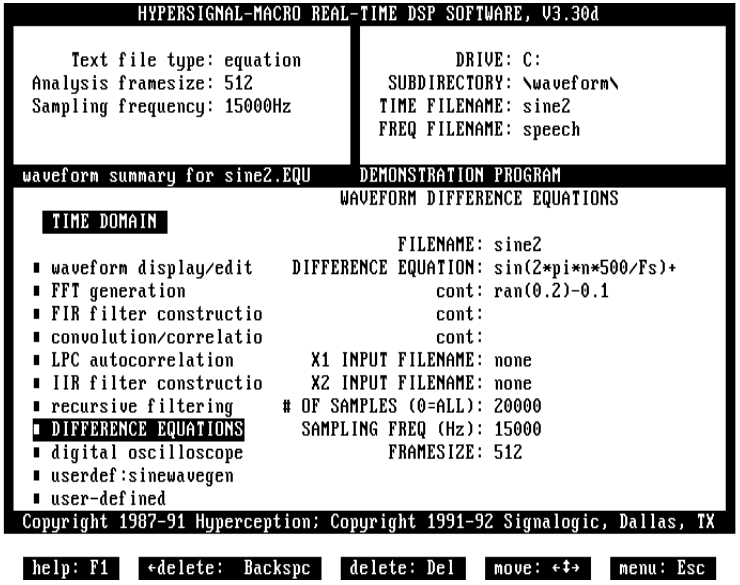


*Figure 8-6*

This example will add random noise with maximum amplitude equal to 10% of the sine waveform's maximum amplitude. In this case the "RAN(0.2)" entry is creating pseudo-random numbers between 0 and 0.2—note the subtraction of 0.1 to center the pseudo-random numbers about zero (-0.1 to 0.1). Figure 8-7 below shows this waveform in the Waveform Display/Edit function (with the FRAMESIZE field=512).

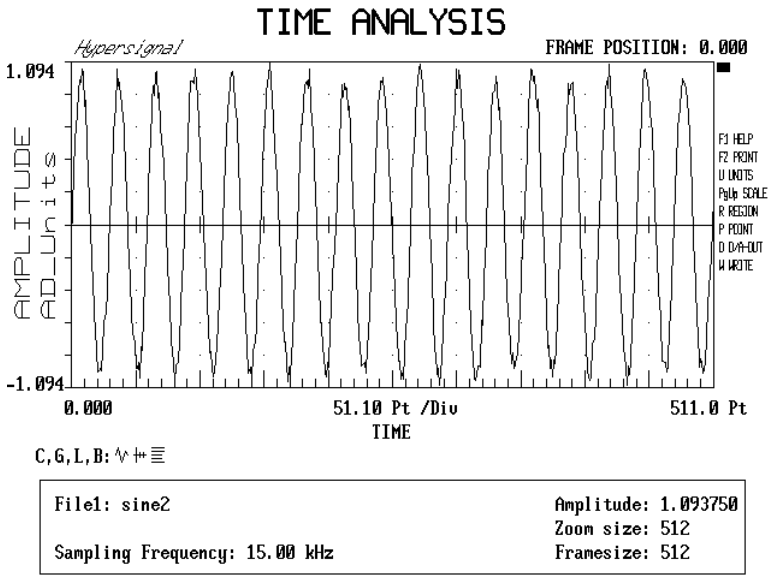


*Figure 8-7*

As a fourth waveform generation example, enter the Difference Equations below menu to create a “dual-tone modulated frequency” (DTMF) waveform that is similar to what you hear when pressing the “7" button on a telephone with touch-tones.

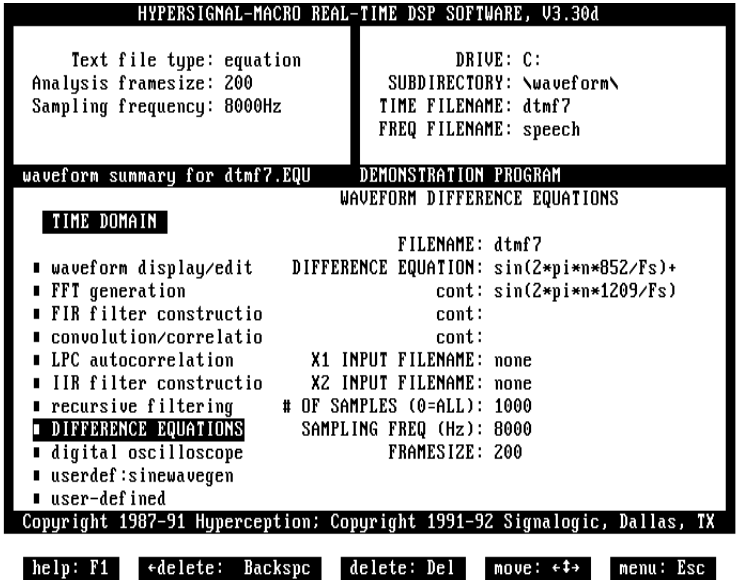


*Figure 8-8*

After allowing the above difference equation to be evaluated, enter the following FFT Generation menu:

FILENAME: dtmf7
FRAMESIZE: 200
FFT ORDER: 9
% FRAMESIZE OVERLAP: 0
WINDOW (R/Hm/Hn/Bl/Bt): R
REAL/IMAG OUTPUT (Y/N): N
PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to 'none' entry). After allowing the above FFT Generation to complete, enter the following Magnitude Display menu:

FILENAME1: dtmf7
FILENAME2:
POWER SPEC (P/M): M
AMP SQUARE (Y/N): N
LOG (N/+/-,N/base,Min): N
Y-LABELS (SCALE,UNITS): 1,Volts

AMP SCALING (TOP,BOT,REF): 0,0,A
FORMAT,PLOT (D/O,B/C/D): D,B
GRID (N/S/D/I): I

Figure 8-9, below, shows the resulting Magnitude Display:

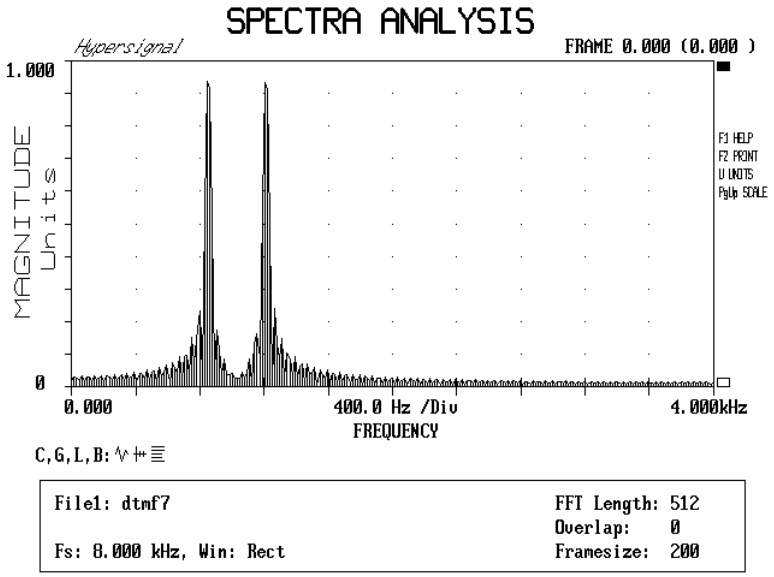


*Figure 8-9*

If you were to build a series of these “dtmfN” waveforms, you could string them together (using either another difference equation or the waveform editor) to simulate a telephone number, then use the Analog Conversion function to execute a stimulus & response operation to test a real-time touch-tone detection algorithm’s response to the number (see chapter 12, STIMULUS & RESPONSE). You could, for example, add previously recorded “narrative” speech into the touch-tone stimulus with the Difference Equations function, and then measure the robustness of the detection algorithm to this type of periodic energy contamination.

## 8.2 Signal Arithmetic

Now that we have used the Difference Equations function to generate waveforms from scratch, let’s use it to manipulate existing waveforms.

As a first example, we will heterodyne the “sine” and “sine2" waveforms created in the previous waveform generation examples. Enter the following Difference Equations menu:

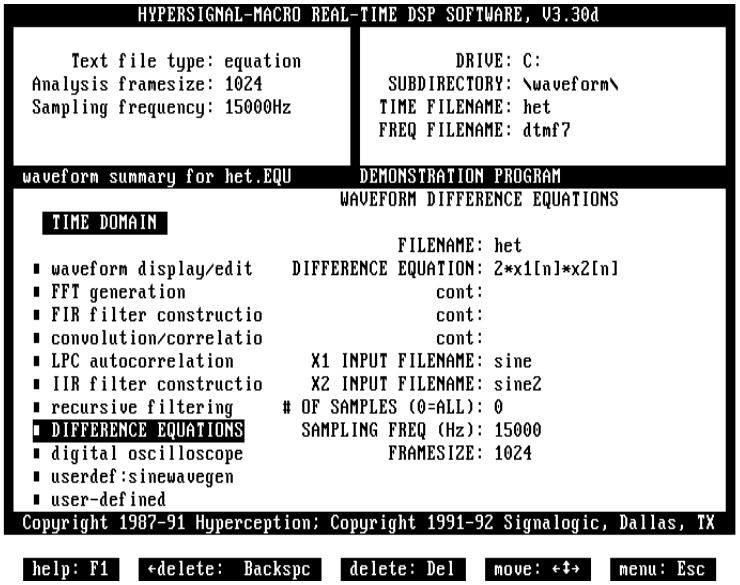


*Figure 8-10*

This will “mix” (multiply) the two waveforms together, causing frequencies of 700 Hz and -300 Hz to appear in the output waveform “het”. This results from the basic trigonometry forumla of

$$2\sin(a)\sin(b) = \cos(a-b) - \cos(a+b)$$

Note the ‘0’ entry in the # OF SAMPLES field, which instructs the program to use all samples from the x1 and x2 waveforms, if referenced in the equation (whichever is longer if both are referenced). To see the results of this operation clearly, we will look in the frequency domain. After allowing the Difference Equation menu shown in Figure 8-10 above to execute, enter the following FFT Generation menu:

FILENAME: het
FRAMESIZE: 1024
FFT ORDER: 10
% FRAMESIZE OVERLAP: 25
WINDOW (R/Hm/Hn/Bl/Bt): Hm
REAL/IMAG OUTPUT (Y/N): N

```
          PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N
```

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to ‘none’ entry). After executing the above FFT Generation, enter the following Magnitude Display menu:

```
                FILENAME1: het
                FILENAME2:
          POWER SPEC (P/M): M
         AMP SQUARE (Y/N): N
      LOG (N/+/-,N/base,Min): N
    Y-LABELS (SCALE,UNITS): 1,AD_units
AMP SCALING (TOP,BOT,REF): 0,0,A
   FORMAT,PLOT (D/O,B/C/D): D,B
          GRID (N/S/D/I): I
```

Note that in figure 8-11 below the -300 Hz frequency actually appears at 300 Hz. Since the total bandwidth that can be displayed for a non-complex signal is from 0 (dc) to the sampling frequency/2, the -300 Hz value must “reflect” about dc and appear at 300 Hz. Note also the “white noise floor” energy that results from the sine2 waveform being made impure by the addition of random noise. Thus, the original “sine” waveform frequency of 200 Hz has been shifted both up and down by the “sine2" waveform frequency of 500 Hz. By applying a digital filter to remove one of these resultant frequencies, the heterodyne equation can serve as a ”frequency shifter", sliding a segment of bandwidth to a new position. When combined with a sampling rate conversion (see section 15.1, Sampling Rate Conversion), this technique can be used to selectively process segments of a waveform’s bandwidth.

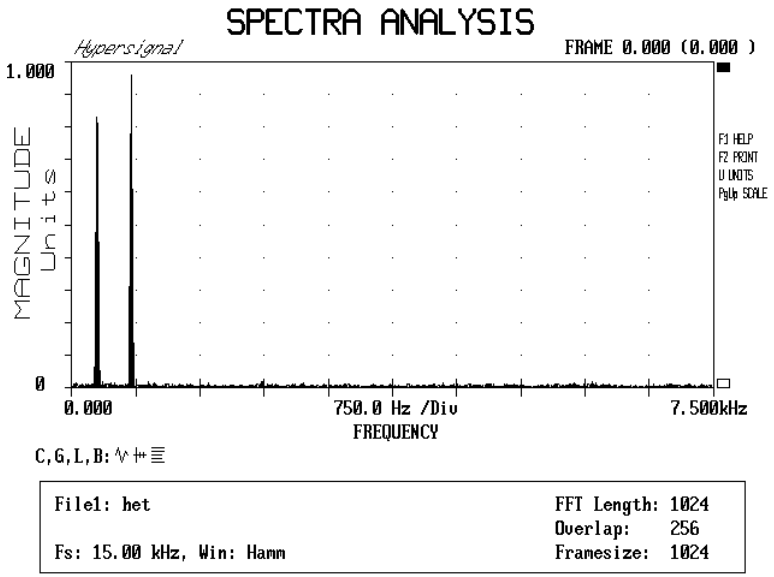


*Figure 8-11*

As a second example, we'll use the Difference Equations function to create the "superposition" of the sine and sine2 waveforms. Enter the following Difference Equations menu:

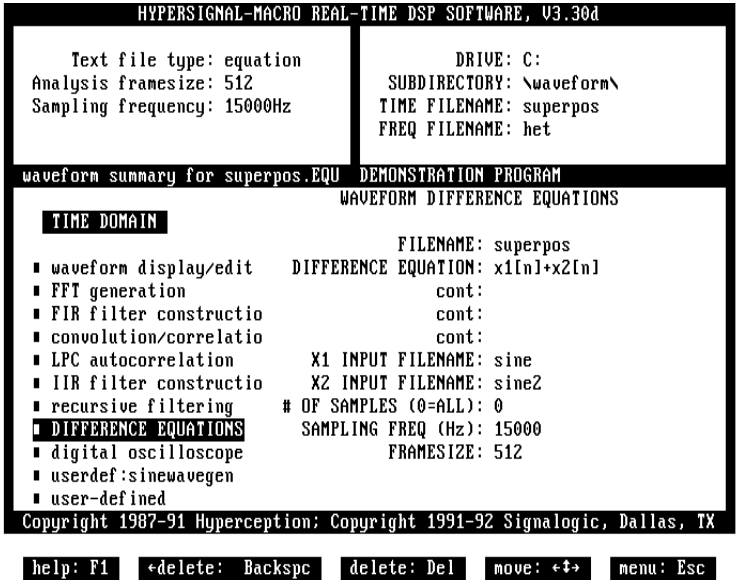


*Figure 8-12*

After evaluating the above difference equation, enter a Waveform Display/Edit menu with the FILENAME1 field set to “superpos”, the FILENAME2 field blank, and the FRAMESIZE field set to 512 to see classic constructive and destructive interference phenomenon in the newly created “superpos” waveform:

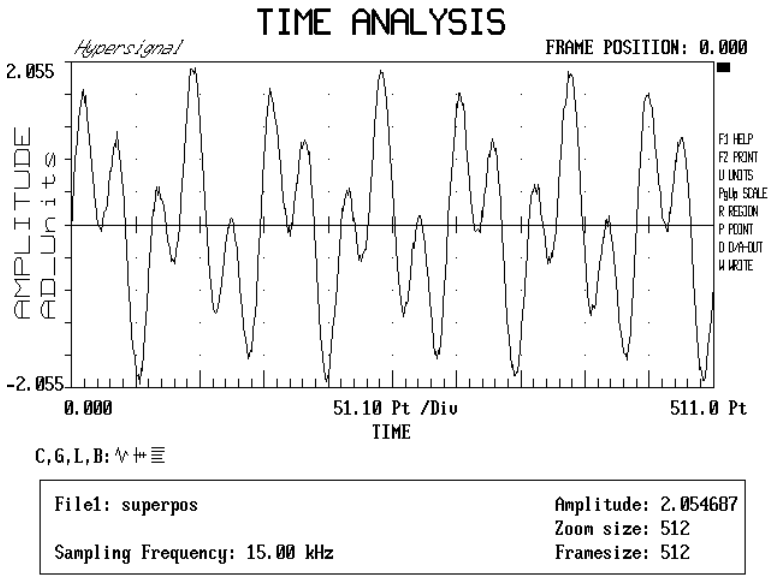


*Figure 8-13*

Remember that you can use the arrow keys (and mouse motion) to position a target window over the waveform, and then the 'Home' key (or middle button on the mouse) to zoom in on the displayed waveform.

As a third waveform manipulation example, enter the following Difference Equations menu:

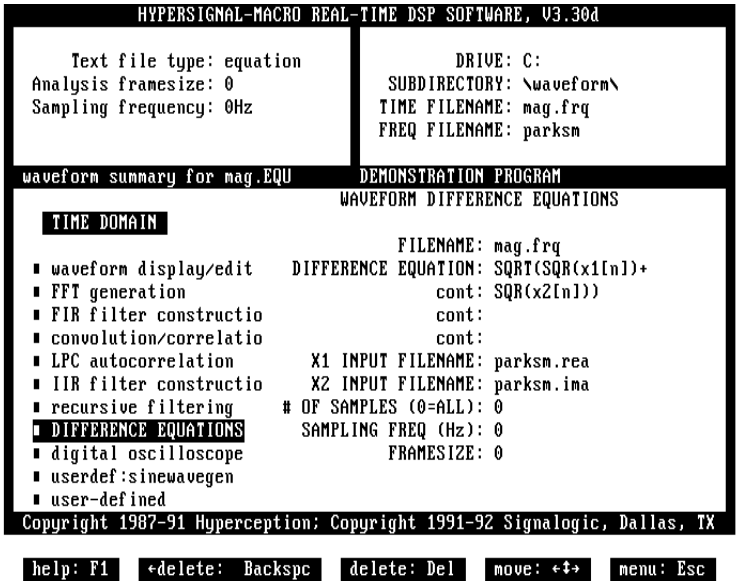


*Figure 8-14*

In this example we assume that the real and imaginary output from a previous FFT Generation operation on the “parksm” FIR filter impulse response is available (see chapter 6, FFT GENERATION). A magnitude waveform will be created that represents the “parksm” FIR filter’s frequency response, and is suitable for display using the Hypersignal Magnitude Display function. Normally this is done automatically by the FFT Generation function but this also makes a good Difference Equations example. Figure 8-15 below shows the Magnitude Display of the Difference Equation-generated magnitude and the FFT Generation-generated magnitude—which, of course, are the same. Note that the “.FRQ”, “.REA”, and “.IMA” extensions had to be entered explicitly in order to override the default extensions (which are “.TIM” to indicate time domain files). Note also the use of ‘0’ entries for the SAMPLING FREQ and ANALYSIS FRAMESIZE fields, which instruct the program to get these values from the x1 or x2 waveforms (x1 if both are referenced in the equation).

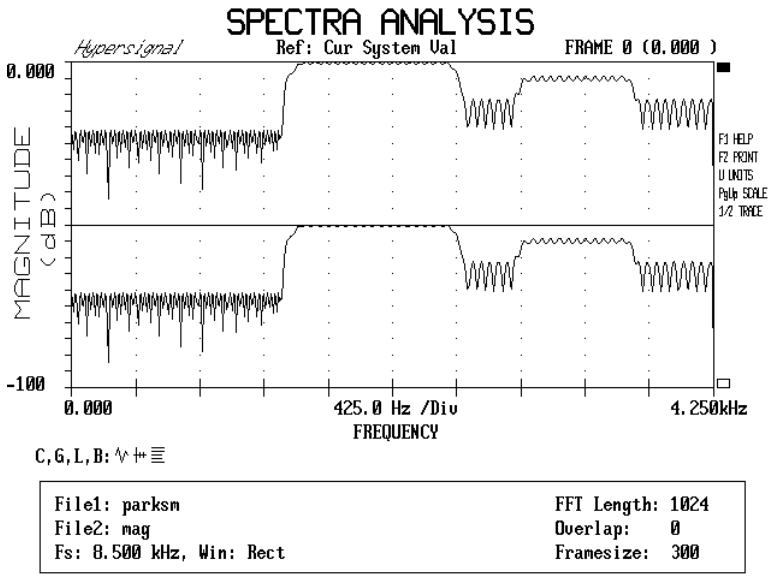


*Figure 8-15*

You can also construct a Hypersignal-compatible phase waveform file using the following difference equation:

y[n]=4096∗((sign(x1[n]-1)/2)∗sign(x2[n])∗pi+
atan(x2[n]/(x1[n]+1e–20)))

The (sign(Real)-1)/2 causes the sign(Imag)∗π term to be added only if the real value is negative; otherwise, nothing is added. The 4096 multiplier is used only to cause the result to be Hypersignal-compatible, allowing results to be compared using the Phase Display function. Note that differences still might exist when imaginary values are exactly equal to zero, but unwrapping the phase should cause identical displays. As with the magnitude example above, you should use '0' entries for the # OF SAMPLES, SAMPLING FREQ, and FRAMESIZE fields, to ensure that all data is processed and attributes inherited from the input waveforms.

It is also possible to "reverse" the above examples, and return to real and imaginary data. Use the following two difference equations:

y[n]=x1[n]∗cos(x2[n]/4096)

y[n]=x1[n]∗sin(x2[n]/4096)

where in both examples, x1 should refer to the “mag.frq” magnitude waveform calculated above, and x2 should refer to the phase waveform.

As a fourth waveform manipulation example, we offer the “boxcar filter” (also known as a running sum), and the “limiter”, which are two relatively common algorithms. To construct a running sum, use the following difference equation:

y[n]=y[n–1]+x1[n]/L–x1[n–L]/L

where ‘L’ should be entered as the length of the running sum, in samples. In the above equation, the y[n-1] term serves as an accumulator, the +x1[n] term adds the “locomotive” sample, and the -x1[n-L] term subtracts the “caboose” sample. Next, we have the limiter. In the example below, all samples in the input waveform with magnitude less than some threshold will be set to zero. Note that both positive and negative samples are affected:

y[n]=u[abs(x1[n])–T]∗x1[n]

where ‘T’ should be entered as a threshold value. Especially note the use of the “unit-step” function in the above equation. Because this function returns either a 0 or 1, it can be used as the threshold “trigger”, to determine when values should be set to zero. In the example, when input waveform values are less than the threshold, input to the unit-step function is less than zero, causing a 0 output, and forcing the output to zero.

For more difference equation examples, see chapter 15, MACRO EXAMPLES, which contains several subsections involving macro language text files that invoke the Difference Equations function during intermediate processing. For detailed information about the Difference Equations function, and a list of all functions, operators, constants, etc., see Users Manual section 4.6.1, Writing Difference Equations.

## 8.3 Sample Manipulation

Now that we have used the Difference Equations function to generate waveforms from scratch and perform signal arithmetic on existing waveforms, let’s use it to move simply waveform samples around.

The examples chosen for this section perform simple but useful sample manipulation tasks. If you understand the basic “Z-shift”, or delay equation

y[n]=x1[n–N]

where N is the number of samples to delay the x1 waveform, then you already understand the examples in this section. Also, the power of the Difference Equations function for controlling samples and sample indexes should become more apparent. Simple functions that make you wonder "how would they do this in the analog domain" can be done easily in the "equation domain". For example, you could reverse the time sequence of a waveform with the following Difference Equations menu:

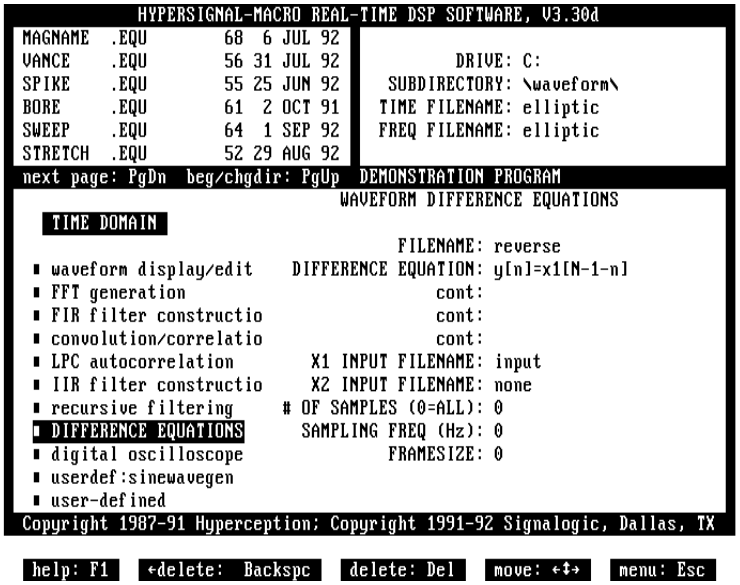


*Figure 8-16*

In the example above, you would need to supply a value in place of "N" which should be the length of, or number of samples in, the input waveform specified in the X1 FILENAME field. "Input" is the input waveform filename; "reverse" will be the output waveform filename. Entry of '0' for # OF SAMPLES, SAMPLING FREQUENCY, and FRAMESIZE will cause the output waveform to have the same values for these parameters as the input waveform. In particular, the '0' entry for # OF SAMPLES will cause all samples of the input waveform to be processed. Alternatively, you could build a simple macro language file that executed the above equation but filled in the equation with a statement similar to

y[n]=x1[@N-1-n]

where N was a declared variable, and was initialized by a "GetHeaderValue(filename, Len)" call, which would return the length of the waveform specified by "filename". The '@N' entry in the macro language file would be evaluated at run-time and replaced by a string representation of the value of N.

As another example, consider an "interpolation" operation, where you must increase the sampling rate of a waveform by replicating every sample N times. After such an operation, a lowpass "smoothing" filter can be applied that completes the transition to a higher sampling rate. The menu below shows a N-times interpolation equation:

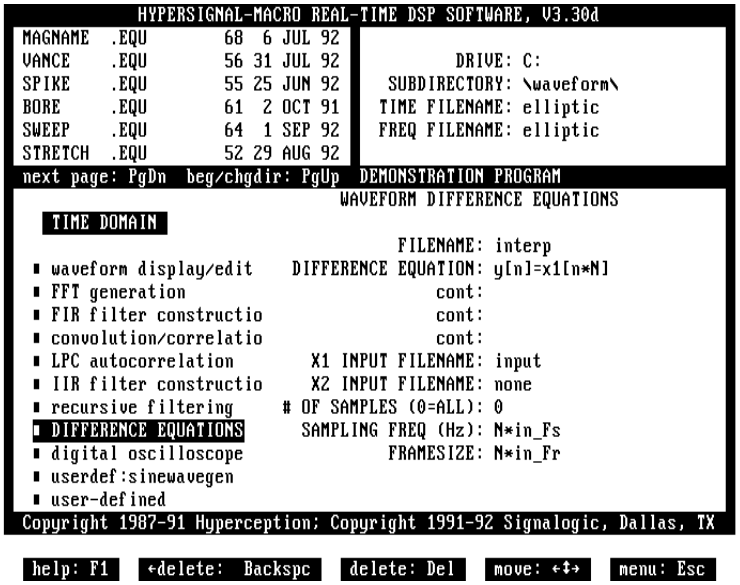


*Figure 8-17*

In the example above, you would need to supply a value in place of "N" which should be the number of samples to replicate. "Input" is the input waveform filename; "interp" will be the output waveform filename. Entry of '0' for # OF SAMPLES indicates that all input waveform samples should be processed. The SAMPLING FREQUENCY and FRAMESIZE fields are shown as 'N*in_Fs' and 'N*in_Fr', which are not valid entries, to imply that these values should be set as N times their counterparts from the input waveform. Setting these values automatically can be done easily using a macro language file, where the input waveform header is accessed

prior to the difference equation, and the appropriate values are placed in the Difference Equations data-entry menu as variables that are evaluated and converted to a text string at run-time. See section 15.1, Sampling Rate Conversion, for excellent examples of this type of difference equation, and using a macro language file to set up the data-entry menu fields.

As a last sample manipulation example, consider the “circular buffer” equation shown below. This equation moves the data in the input waveform an arbitrary number of samples (specified by “N” in the equation below) towards time = zero. Data that would enter negative time instead “wraps around” to the end of the waveform. The menu below shows the circular buffer equation:

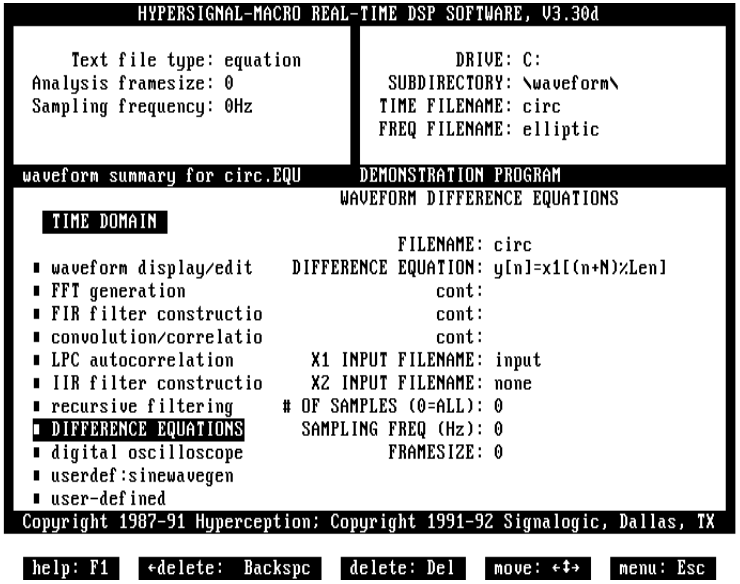


*Figure 8-18*

In the example above, you would need to supply values in place of “N” and “Len”, which would be the number of sample shifts and the length of the waveform, respectively. Note the modulus operation (indicated by the ‘%’ operator) that causes the index to wrap back to zero when n+N equals Len. “Input” is the input waveform filename; “circ” will be the output waveform filename. Entry of ‘0’ for # OF SAMPLES, SAMPLING FREQUENCY, and FRAMESIZE will cause the output waveform to have the same values for these parameters as the input waveform. In par-

ticular, the ‘0’ entry for # OF SAMPLES will cause all samples of the input waveform to be processed. Alternatively, you could build a simple macro language file that executed the above equation but filled in the equation with a statement similar to

```
y[n]=x1[(n+#1)%@Len]
```

where #1 was a Macro Manager menu input parameter that specified the amount of shift (in samples), and “Len” was a declared variable, which was initialized by a “GetHeaderValue(filename, Len)” call that returned the length of the waveform specified by “filename”. The ‘@Len’ entry in the macro language file would be evaluated at run-time and replaced by a string representation of the value of Len. See chapter 15, MACRO EXAMPLES, for many examples of macro language source files that invoke the Difference Equations function.

*this page intentionally left blank*

# 9.0 DESIGNING FILTERS

This chapter of the tutorial presents a walk-through of both "tolerance-region" design for classical filters and "graphical filter editor" design of arbitrary filters. Although this chapter contains several easy-to-explain examples, for detailed information you should refer to Users Manual section 4.8, Designing Filters in Hypersignal.

"Classical" refers to design methods based on pre-determined mathematical formulae that will produce a filter with certain characteristics and shape once the input parameters to the mathematical formula have been specified. "Arbitrary" refers to design methods that allow the user to specify the filter shape and characteristics first, then work backwards to get the necessary parameters to create such a filter. Classical methods are relatively easy to implement and can produce guaranteed characteristics such as monotonically decreasing stopband, flat passband, etc. depending on the method. Arbitrary methods are iterative and thus require computer-aided calculation—they depend on achieving some mimimum level of error in approximating the desired response in order to be deemed successful. With arbitrary methods, nothing is guaranteed—not the time taken to arrive at an answer or the final form of the answer, although in recent years methods have been developed that have proven fairly reliable when used carefully.

Both classical and arbitrary design methods can be used to create "standard" filter types: lowpass, highpass, bandpass, and bandstop. However, only arbitrary methods can be used create "non-standard" types such as multiband, irregularly-shaped passband, curves, etc.

Also, both classical and arbitrary methods can be used to create FIR and IIR filters. FIR stands for "finite impulse reponse" and refers to filters with transfer functions that have no "poles", only "zeros". IIR stands for "infinite impulse response" and refers to filters with transfer functions that have both poles and zeros. Since poles are in the denominator of a transfer function, they imply feedback; feedback, of course, is previous output being used as current input, which theoretically allows the transfer function to provide infinite output even after external input is removed. Obviously, in practice IIR filters are only useful if their transfer function does indeed "die out", or decay to zero when input is removed—if not, then the filter is said to be "unstable". FIR filters, without any poles in

their transfer function, can only provide output for a finite period of time after input is removed. Generally speaking, FIR and IIR filters have the following two major tradeoffs:

- Given enough coefficients, FIR filters can achieve any degree of "goodness" in terms of passband roll-off, ripple, stopband attenuation, etc. without becoming unstable. An IIR filter can always achieve the same level of "goodness" as its FIR counterpart with less coefficients, but can be very difficult to make stable in some cases. Since the number of coefficients translates into computation time, the choice of which type of filter to use is often based upon how fast the filter must execute, especially for real-time applications.

- FIR filters provide exact control over the phase of the output waveform; generally, "linear phase" FIR filters are used that provide a constant amount of delay for all frequencies (group delay) being filtered. IIR filters can do some nasty things to the phase of the filtered waveform; in some cases where group delay is very important, a second, "allpass" filter must be applied that equalizes the phase distortions of the first filter.

## 9.1 Classical IIR Filters

First, we will design some classical IIR filters, starting with a Butterworth lowpass filter. Enter the following IIR Filter Construction menu:

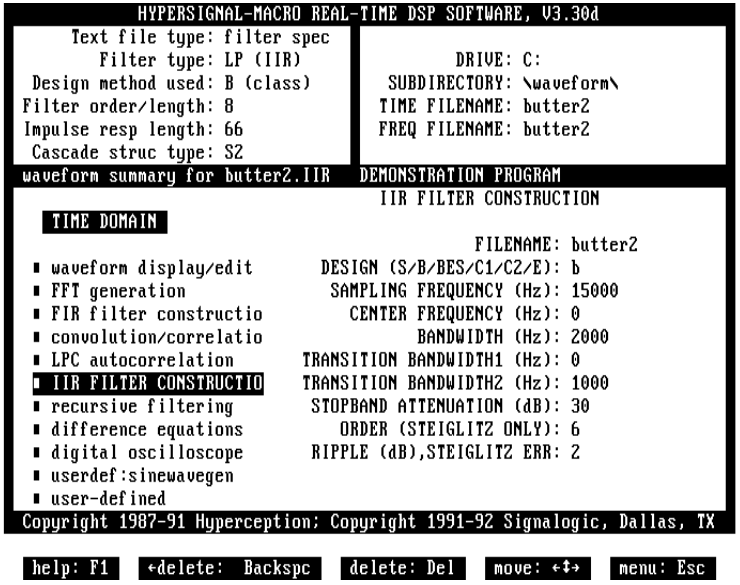


*Figure 9-1*

A CENTER FREQUENCY value of zero basically tells the program that this will be a lowpass filter; the value of BANDWIDTH then specifies the width of the passband region, starting from zero. In this case, the TRANSITION BANDWIDTH1 field doesn't matter, since the CENTER FREQUENCY value of zero causes it to be ignored. The TRANSITION BANDWIDTH2 field, however, is used to control the slope, or "roll-off" of the lowpass filter, i.e. the width of the frequency region in which the filter response transitions from passband to stopband. The STOPBAND ATTENUATION field specifies how far "down" the stopband will be, and the RIPPLE field controls the amount of ripple in the passband area. For Butterworth and Chebyshev I filters, which are flat in the passband, the RIPPLE field actually specifies the maximum amount of dB down the response can be at the edges of the passband, since this is the only "ripple" present in the passband. The ORDER and STEIGLITZ ERR fields apply only to filters using the Steiglitz arbitrary design method (see section 9.2, Arbitrary IIR Filters, below).

While the cursor is positioned on the last field, pressing the 'Enter' key (or any button on the mouse) will cause the program to advance to the "graphical filter design" display, where the filter specification "tolerance regions" will be outlined graphically. In order to meet specifications, the

filter response must fall completely within the tolerance regions. If the graphical outline looks reasonable to you, press the ‘C’ key to begin construction of the filter. At this point a prompt will appear that lists the order required for different filter design methods and offers you the opportunity to switch methods if desired. Press ‘C’ to continue without changing methods.

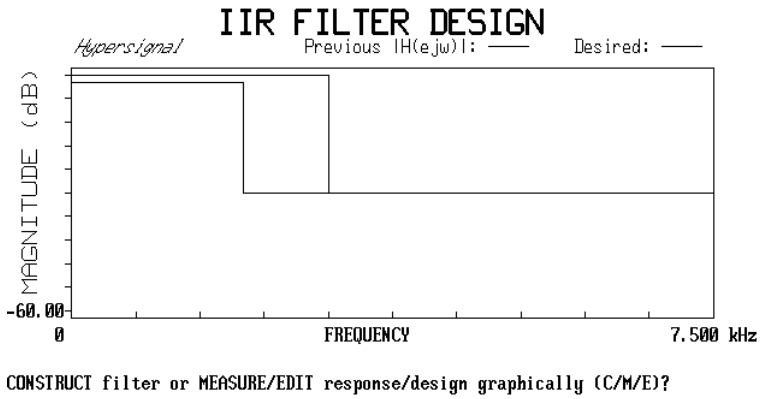


*Figure 9-2*

After the IIR Filter Construction function has finished, enter the following FFT Generation function menu, using the newly created “butter2" impulse response as input:

| | |
|---|---|
| FILENAME: | butter2 |
| FRAMESIZE: | 66 |
| FFT ORDER: | 10 |
| % FRAMESIZE OVERLAP: | 0 |
| WINDOW (R/Hm/Hn/Bl/Bt): | R |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been

disabled (set to ‘none’ entry). After executing the above FFT Generation, return to the IIR Filter Construction menu:

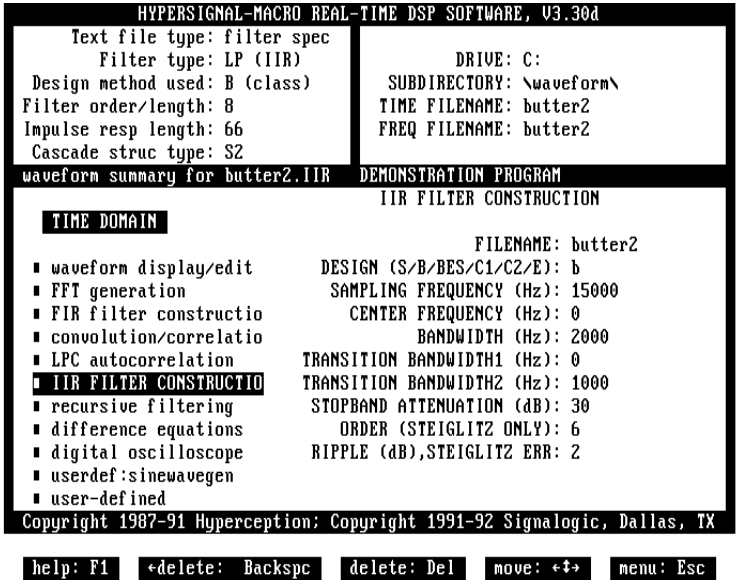


*Figure 9-3*

Note that nothing should be changed from the original design. After advancing to the graphical design menu, press the ‘E’ or ‘M’ keys to bring up the full graphical design Edit/Measurement display. You should see the actual response displayed along with the tolerance regions, usually in a different color. If the actual response is contained within the tolerance regions, the filter has met specification. You can use the arrow keys (or mouse motion) to move the crosshair cursor around to make measurements on the response. The current cursor position is updated continuously in dB and Hz at the lower right of the display. Note that you can also flip between log and linear horizontal and vertical axes by repeatedly pressing the ‘L’ key; pressing the ‘G’ key will turn on and off a grid intended to aide in measurement. Pressing the ‘PgUp’ key will “zoom-in” vertically; pressing the ‘PgDn’ key will “zoom-out” vertically, allowing you to see more of the actual response.

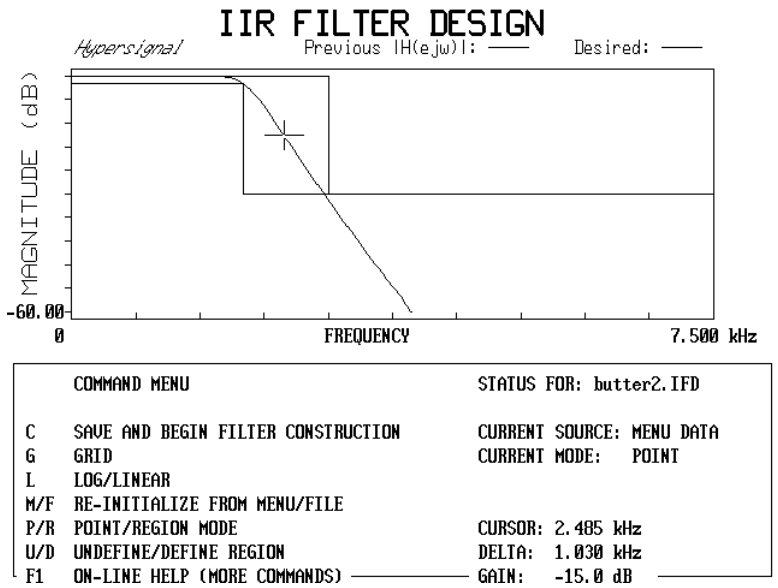


*Figure 9-4*

Following are menu entries for other classical IIR filter types. The design procedure is the same as above for the filter “butter2", so we only present the menus.

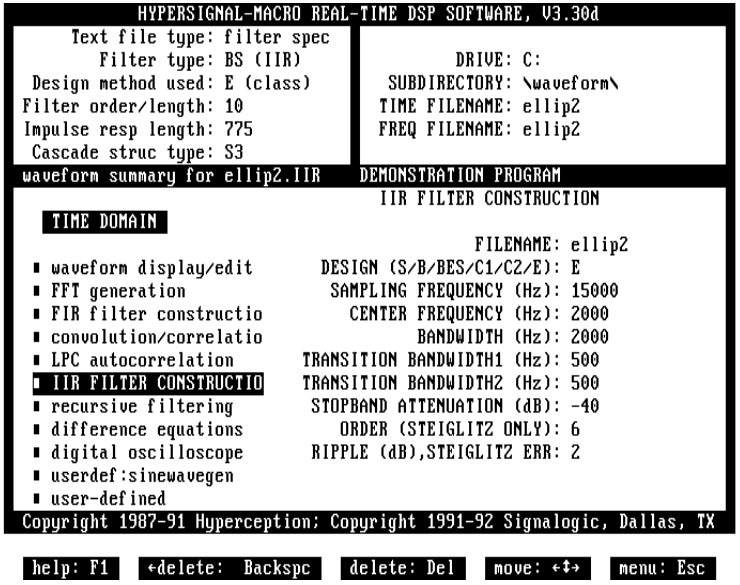


*Figure 9-5*

Above is the menu for an elliptic bandstop design. A non-zero CENTER FREQUENCY value basically tells the program that this will be a bandpass (or bandstop) filter; the value of BANDWIDTH then specifies the width of the passband (or stopband) region to be centered about the value of CENTER FREQUENCY. In the case of a bandpass or bandstop filter, the smaller of the TRANSITION BANDWIDTH1 and TRANSITION BANDWIDTH2 fields controls the slope, or “roll-off” of the filter, i.e. the width of the frequency region in which the filter response transitions between passband and stopband. The STOPBAND ATTENUATION field specifies how far “down” the stopband will be on either side of the passband region. Below is the resulting graphical display (after design and FFT Generation, as in previous examples):

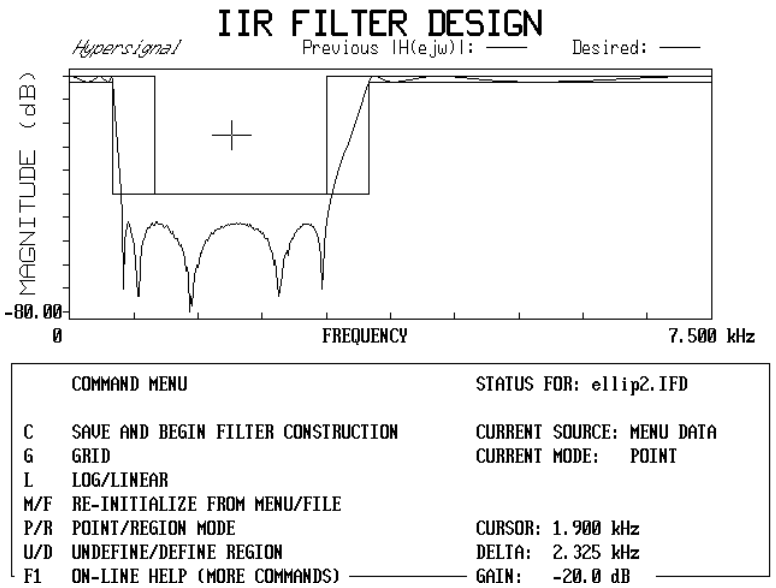


*Figure 9-6*

*Note that a positive value in the STOPBAND ATTENUATION field would specify a bandpass design instead*, as in the following Chebyshev II bandpass design:

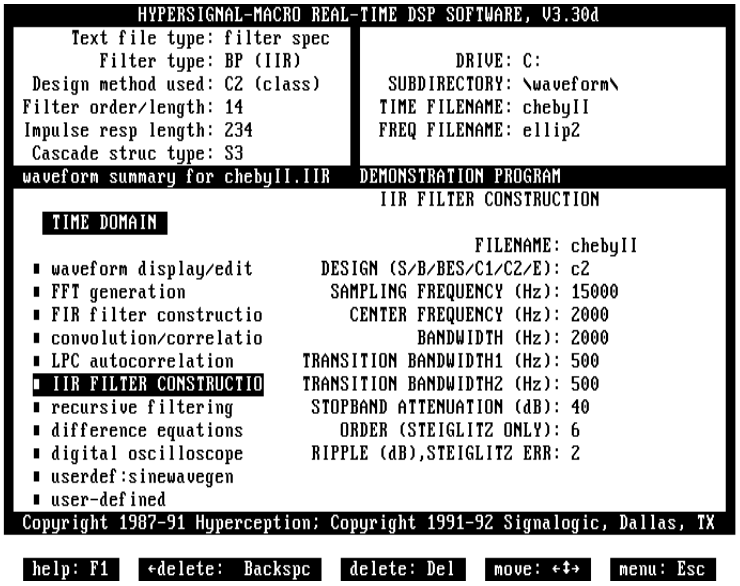


*Figure 9-7*

Below is the resulting graphical display (after design and FFT Generation, as in previous examples):

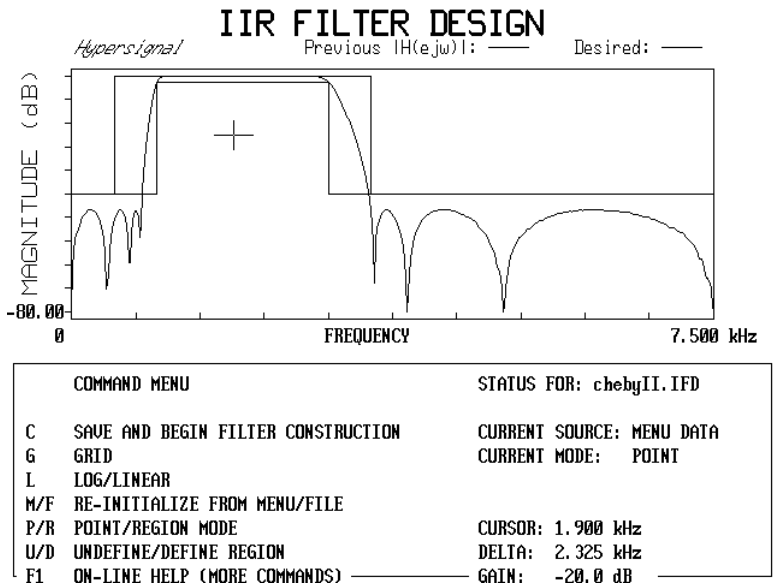


*Figure 9-8*

Note the ripple in the stopband areas and the lack thereof in the passband, which is characteristic of the Chebyshev II design method. Below is a Chebyshev I highpass design:

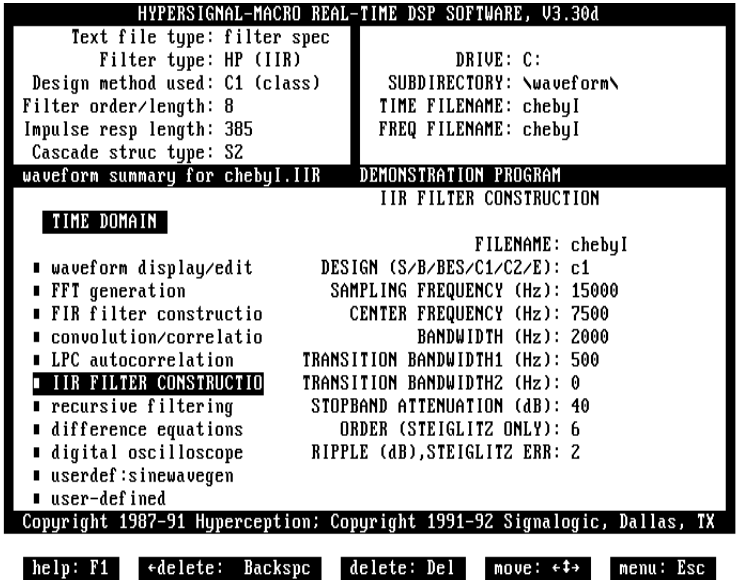


*Figure 9-9*

A CENTER FREQUENCY value of half the sampling frequency basically tells the program that this will be a highpass filter; the value of BANDWIDTH then specifies the width of the passband region, ending at half the sampling frequency. In this case, the TRANSITION BANDWIDTH2 field doesn't matter, since the CENTER FREQUENCY value of half the sampling frequency causes it to be ignored. The TRANSITION BANDWIDTH1 field, however, is used to control the slope, or "roll-off" of the filter, i.e. the width of the frequency region in which the highpass filter response transitions from stopband to passband.

## 9.2 Arbitrary IIR Filters

Now lets suppose that you want to design something non-standard and still use an IIR implementation. For this purpose Hypersignal offers the Steiglitz arbitrary design method. In the example below, we will use a specification file already present on your WAVEFORM subdirectory, to save you the time of creating an arbitrary response:

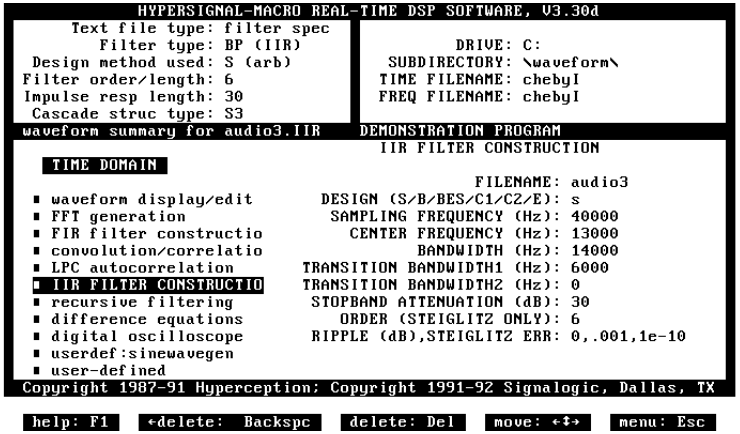


*Figure 9-10*

Note that in the case of Steiglitz designs, the ORDER field is important, since the order of the filter must be specified beforehand. Of course, this is backwards from classical methods which tell you the order based on the other menu parameters. But, as with all arbitrary filters, every parameter is input; the big unknown is whether the design method will converge and achieve the minimum error. In the example above, the ripple has been set at 0 dB, the "first-pass" minimum error has been specified as $10^{-3}$, and the final minimum error value has been specified in the STEIGLITZ ERR field as $10^{-10}$.

**POWER NOTE: In the case of Steiglitz arbitrary filter design, both error values represent the minimum error necessary to obtain convergence, with the first value being used for "initial convergence", and the second for convergence thereafter. This two-level technique is used because more often than not the Steiglitz method must invert unstable poles and zeros outside the Z-domain unit circle after initial convergence, which then results in lower error values for the next convergence. Thus, two-tier entry can avoid an endless search for first convergence that meets a very low minimum error, and instead apply the low minimum error only after a basic pole-zero pattern has been established. See Users Manual section 4.8.1.1, Using the Filter Construction Data-Entry, for detailed information.**

While the cursor is positioned on the last field, pressing the 'Enter' key (or any button on the mouse) will cause the "graphical filter design" display to appear (see Figure 9-11, below; use the 'G' key to get the solid grid lines instead of intersected grid points). Unlike classical filter designs, you will see an outline of the desired filter response instead of tolerance regions. The difference is substance as well as form: because this is an arbitrary filter you can edit the desired response. First press the 'E' or 'M' keys to bring up the full Edit/Measurement menu. Then, press the 'Home' key (or middle button on the mouse) to "tack" the cursor to the desired response. Now use the arrow keys or move the mouse around to change the desired response. Pressing the 'Home' key (or middle mouse button) again will "untack" the cursor; you will be able to move freely once more without altering desired response points. Although this may seem "slick", and can be useful for quick concept verification, we actually recommend creating the desired response via ASCII "design-goal" file, which goes by the filter name and a ".IFD" extension (in this case "audio3.IFD"). This file is read (if pre-existing) and written by Hypersignal whenever you work with arbitrary filter designs. Please consult Users Manual sections 4.8.2, Designing Arbitrary Filters in Hypersignal, and 4.8.3.2, Filter Specification File Format, for detailed information on creating/modifying arbitrary filter design files and the format of these files.

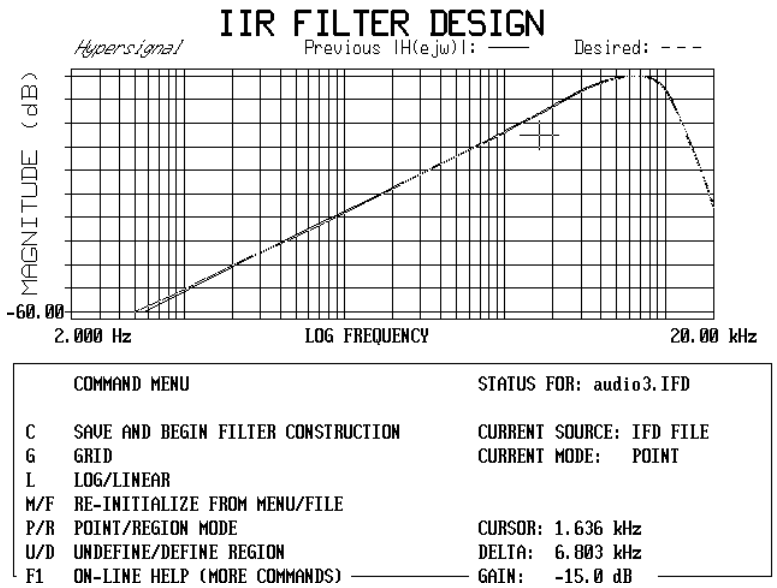


*Figure 9-11*

For now though, let's restore the original "audio3" shape after your editing experiment. Press the 'F' key, which will re-initialize the display from the current design-goal file. Now, for the heck of it, press the 'M' key, which will re-initialize the display from the menu parameters. Now press the 'F' key again. When you move the crosshair cursor around, note that its position continuously updates in dB and Hz at the lower right of the display. Now press the 'R' key to go into "region" mode. In this mode you are moving a box around instead of the crosshair cursor. In region mode you can "delete" and "insert" editing points, because the Steiglitz method relies on non-linear "density" of frequency points in order to concentrate error minimization in some areas (higher density) at the expense of others (lower density). Pressing the 'P' key will return to "point" mode, where the crosshair cursor is displayed. Again, we recommend that for serious work the entire response, both amplitude and frequency values, be stored directly to an ASCII specification (design-goal) file via your own external program—Hypersignal can then read this file and go from there (see Users Manual section, 4.8.3.2, Filter Specification File Format, for information on the format of design-goal files).

To create (or re-create) the "audio3" filter, press the 'C' key. Filter construction will now begin, but the time necessary to complete an arbitrary

design will depend greatly on the speed of your system. A math coprocessor will greatly speed up the overall design time, from 4 to 6 times. A fast system (25 MHz, 33 MHz) would also help. Depending on the speed of your system, the "audio3" design may take anywhere from a few minutes to a few hours to complete.

**POWER NOTE: To achieve the IIR design you need, you may have to experiment with different frequency point densities in sensitive areas, especially transition band areas, and with the Steiglitz ORDER field, as well. This may require multiple iterations through the design process: edit the response (or change the ASCII design goal file directly), re-run the Steiglitz method, FFT the response, measure the response; repeat. The Users Manaul contains extensive information on this subject, including suggestions on frequency density for Steiglitz design (see Users Manual section 4.8.2, Designing Arbitrary filters in Hypersignal). Generally, we have found it best to start with a Steiglitz ORDER of 6, and TRANSITION BANDWIDTH values of zero ("brickwall", which initializes transition densities to high setting) and work from there. Let us say beforehand, though, that depending on the filter type, it can be a frustrating and tedious process to arrive at a design-goal that works reliably. However, it is also probably fair to say that this is generally true for all IIR arbitrary design methods, since this is still one of the "black art" areas of digital filtering.**

## 9.3 Classical Fir Filters

Because the FIR Filter Construction function is very similar to the IIR Filter Construction function, we are only going to show a bandpass example for the Kaiser Window classical design method—for lowpass, highpass, and bandstop examples, see section 9.1, Classical IIR Filters, above. Similar to the previous "chebyII" IIR design, below is a Kaiser-Window classical bandpass design:

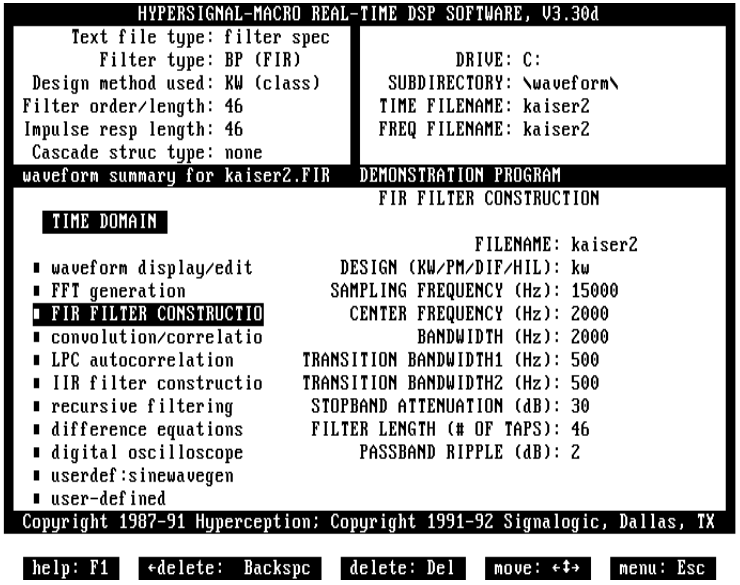


*Figure 9-12*

Note that the menu differences between IIR and FIR designs are slight. The most important is the FILTER LENGTH field, which, for IIR designs is called “ORDER” and is only used for Steiglitz arbitrary designs. For FIR designs, however, the filter length is a parameter relevant to all filter types and under direct user control, although the program will estimate the optimum length of the filter for you given the other menu parameters. Again, after advancing to the graphical filter design menu, you will see tolerance regions that outline the filter’s frequency specifications. The final filter response must fall completely within the tolerance areas in order for the filter to meet specifications. Press the ‘C’ key to begin construction of this filter. Answer ‘Y’ to the length estimation prompt to use the program’s estimate (based on empirical formulae and look-up tables); answer ‘N’ to override the estimate and use your entered length. If the entered length is within ⌠1 of the estimated length, no length prompt will appear.

**POWER NOTE: Generally, FIR length estimations will be within 1% to 3% of the optimal length, and are always adjusted for any odd-even length requirement (filters with non-zero gain at half-sampling frequency, such as highpass and bandstop, should be odd, Hilbert transform filters odd, and differentiators even). For FIR filters, passband ripple values less than 0.5 dB or larger than 3 dB are especially hard to land on exactly and may neccesitate some manual length adjustment.**

After the FIR Filter Construction function has finished, perform an FFT on the “kaiser2" impulse response using the following settings (to which the FFT Generation should initialize to automatically):

| | |
|---|---|
| FILENAME: | kaiser2 |
| FRAMESIZE: | 46 |
| FFT ORDER: | 10 |
| % FRAMESIZE OVERLAP: | 0 |
| WINDOW (R/Hm/Hn/Bl/Bt): | R |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (‘none’ entry). After executing the FFT Generation, return to the FIR Filter Construction menu.

Note that nothing should be changed from the original design. After advancing to the graphical design menu, press the ‘E’ or ‘M’ keys to bring up the full graphical design Edit/Measurement display. You should see the actual response displayed along with the tolerance regions, usually in a different color, similar to Figure 9-13 below:

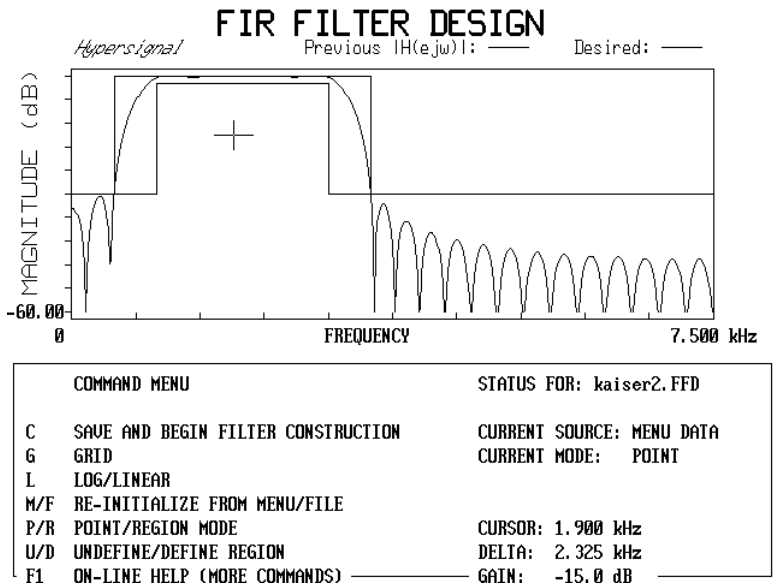


*Figure 9-13*

If the actual response is contained within the tolerance regions, the filter has met specification. You can use the arrow keys (or mouse motion) to move the crosshair cursor around to make measurements on the response. The current cursor position is updated continuously in dB and Hz at the lower right of the display. Note that you can also flip between log and linear horizontal and vertical axes by repeatedly pressing the 'L' key; pressing the 'G' key will turn on and off a grid intended to aide in measurement.

## 9.4 Arbitrary FIR Filters

Now let's design something non-standard using an FIR implementation. For this purpose Hypersignal offers the Parks-McClellan arbitrary design method. In the example below, we will create something totally from scratch:

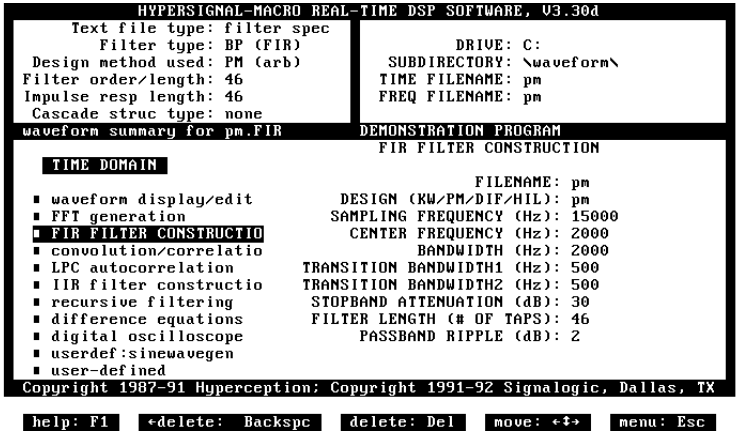


*Figure 9-14*

Note that we have started off with the same specifications (other than the DESIGN method field) as in the “kaiser2" classical example above; however, we will use the arrow keys (or mouse) to further edit the desired response.

Also note that in the case of non-standard arbitrary designs, the FILTER LENGTH field takes on a new importance. For standard lowpass, bandstop, etc. filters, Hypersignal will give you an estimate for FILTER LENGTH that will come very close to (if not right on) the optimum length for the FIR filter given the other menu parameters. However, for designs that are further edited, or whose response is directly entered by ASCII design-goal file, Hypersignal’s estimate is no longer valid, and it is up to you to either relax or strengthen the FILTER LENGTH value as necessary. You may have to iterate through the filter design process a few times to arrive at a FILTER LENGTH value that just meets the specifications. In some ways, the FILTER LENGTH acts as a kind of “reverse mimimum error” control over the final response: the larger the value, the less error between what you ask for and what you get.

While the cursor is positioned on the last field, pressing the ‘Enter’ key (or pushing any button on the mouse) will cause the “graphical filter design”

display to appear. Unlike classical filter designs, you will see an outline of the desired filter response instead of tolerance regions. The difference is substance as well as form: because this is an arbitrary filter you can edit the desired response.

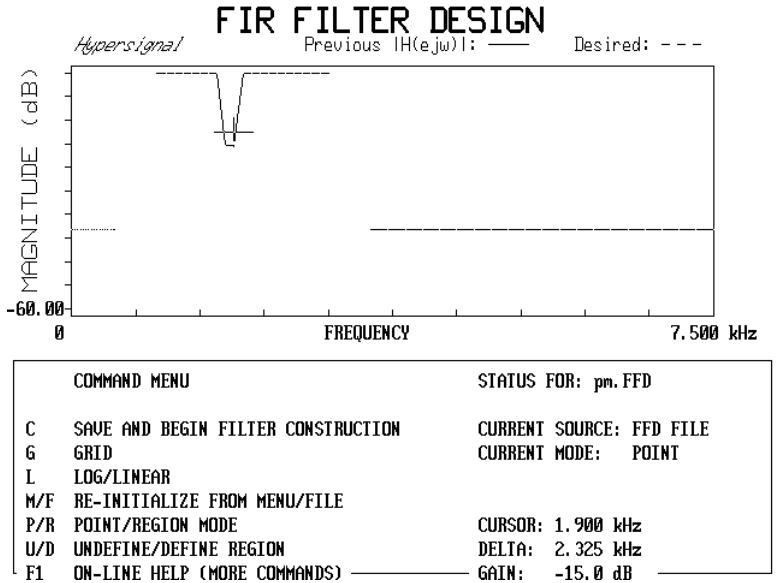


*Figure 9-15*

First press the ‘E’ or ‘M’ keys to bring up the full Edit/Measurement menu. Then, press the ‘Home’ key (or middle button on the mouse) to “tack” the cursor to the desired response. Now use the arrow keys or move the mouse around to change the desired response. Again pressing the ‘Home’ key (or middle mouse button) , you will “untack” from the waveform, and can move freely once more without altering desired response points. Although this may seem “slick”, and can be useful for quick concept verification, we actually recommend creating the desired response via ASCII “design-goal” file, which goes by the filter name and a “.FFD” extension (in this case “pm.FFD”). This file is read (if pre-existing) and written by Hypersignal whenever you work with arbitrary designs. Please consult Users Manual sections 4.8.2, Designing Arbitrary Filters in Hypersignal, and 4.8.3.2, Filter Specification File Format, for detailed information on creating/modifying arbitrary filter design files and the format of these files. In this example though, you are encouraged to draw any picture you want. We would suggest that you make your design some-

what practical, say a two or three-band filter, or change the bandpass to have different slopes on each side, since the Parks-McClellan method can fail to converge if the design is complicated enough or the FILTER LENGTH value short enough. When you move the crosshair cursor around, note that its position continuously updates in dB and Hz at the lower right of the display.

**POWER NOTE: Please note the "undefined" areas in any new arbitrary filter design initialization. These represent the transition band areas, and show no design-goal points to edit. This is done to allow the Parks-McClellan method to produce optimum designs for standard filters such as lowpass, bandstop, etc. By pressing the 'R' key, you will enter "region" mode, and instead of a crosshair cursor, you will see a target window, or box. By placing the target window around an undefined area, and pressing the 'D' key, you can "define" the region, which reinstates the editing points in the region. By pressing the 'P' key, you will return to "point" mode, in which can edit the newly-defined area as necessary.**

Once you are satisfied with your creation, press the 'C' key. You should see a prompt telling you that the estimated length is other than 46 taps (which we entered in the above menu). Answer 'N' to this prompt—the idea here is to find the number of taps that will allow accurate reproduction of your "filter art", yet not take too long to converge. Filter construction will now begin, but the time necessary to complete this arbitrary design will depend greatly on the speed of your system. A math coprocessor will greatly speed up the overall design time, from 4 to 6 times. A fast system (25 MHz, 33 MHz) would also help. Depending on the speed of your system, the "pm" design may take anywhere from a few minutes to a few hours to complete.

After the FIR Filter Construction function has finished, perform an FFT using the following FFT Generation menu, which should represent default settings:

```
                FILENAME: pm
               FRAMESIZE: 46
               FFT ORDER: 10
  % FRAMESIZE OVERLAP: 0
  WINDOW (R/Hm/Hn/Bl/Bt): R
REAL/IMAG OUTPUT (Y/N): N
```

PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled ('none' entry). After executing the FFT Generation, return to the FIR Filter Construction menu.

Note that nothing should be changed from the original design. After advancing to the graphical design menu, press the 'E' or 'M' keys to bring up the full graphical design Edit/Measurement display.

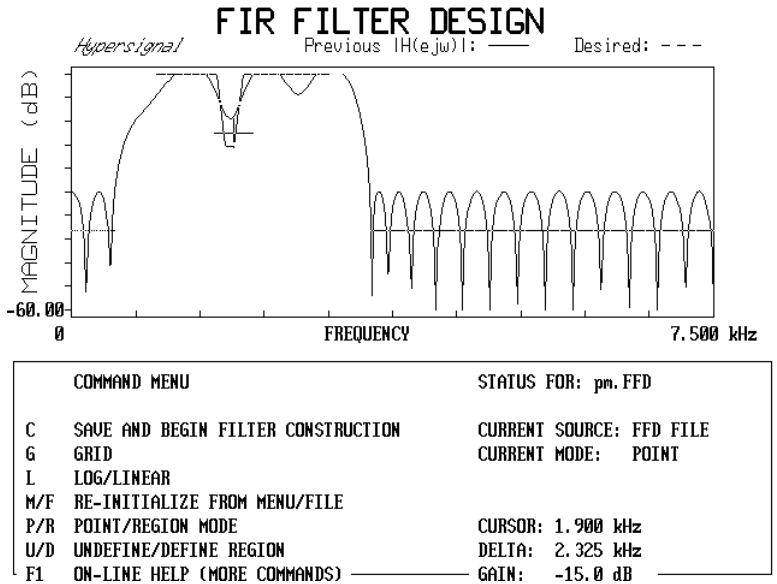


*Figure 9-16*

You should see the actual response displayed along with the desired response, usually in a different color. How close did it come? In the tutorial example, not very. However, by eliminating "un-defined" regions and increasing the number of taps, dramatic improvement is possible. Figure 9-17 below shows the above example with no un-defined regions and number of taps increased to 100, after an FFT Generation similar to the previous listing above. You can also try different PASSBAND RIPPLE values to alter the outcome.

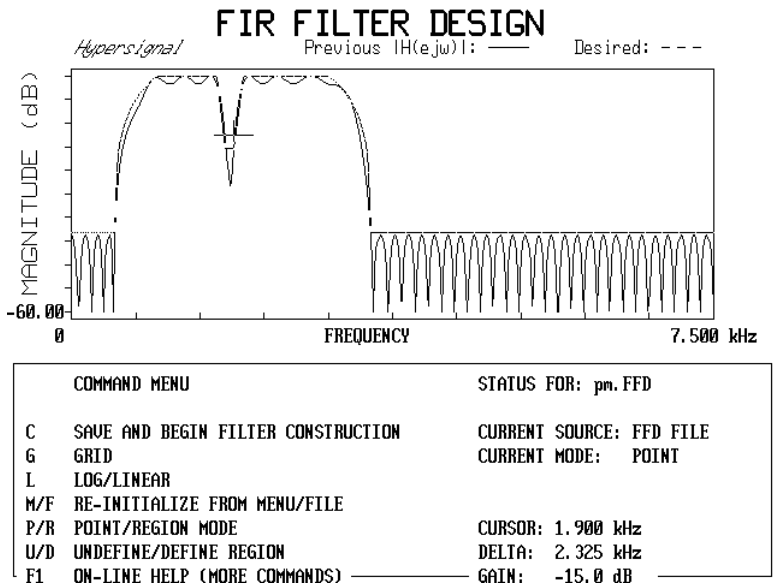


*Figure 9-17*

You can use the arrow keys (or mouse motion) to move the crosshair cursor around to make measurements on the response. The current cursor position is updated continuously in dB and Hz at the lower right of the display. Note that you can also flip between log and linear horizontal and vertical axes by repeatedly pressing the ‘L’ key; pressing the ‘G’ key will turn on and off a grid intended to aide in measurement.

*this page intentionally left blank*

# 10.0 APPLYING FILTERS

This chapter in the tutorial considers three different methods for applying digital filters in Hypersignal: recursive filtering, convolution and real-time “snap-in” filtering. Examples of both applying IIR and FIR filters both on a file-basis (non-real time simulation of filters) and on a real-time basis, where the filter coefficients are downloaded to the current DSP board, are covered.

## 10.1 Recursive Filtering

Enter the following Recursive Filtering menu to apply the “butter2" IIR filter we designed earlier in section 9.1, Classical IIR Filters:

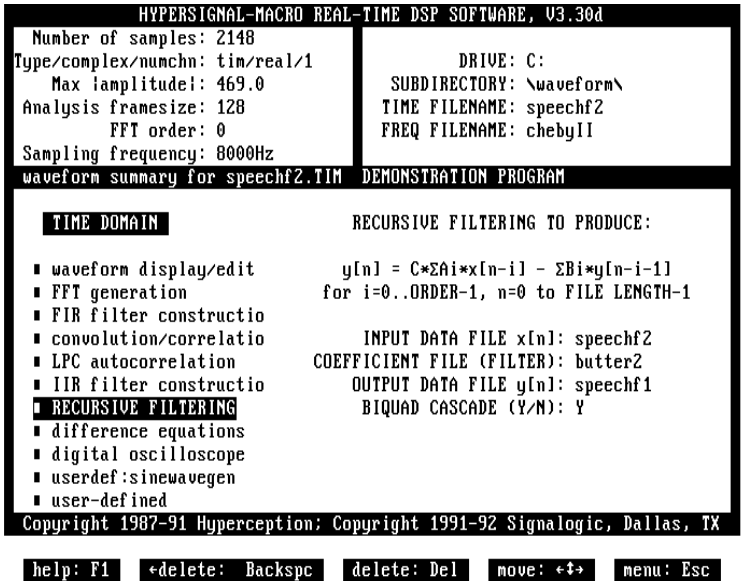


*Figure 10-1*

This example will take the “speech” waveform (should already be on your WAVEFORM subdirectory) and filter it with the “butter2" IIR coefficient file, producing the ”speechf1" waveform as output. The BIQUAD CASCADE field should normally be set to ‘Y’, unless you are using an IIR co-

efficient file of your own creation that is not in factored form (see Users Manual section 4.8.3.1, Filter Coefficient File Format, for more information on creating your own filter coefficient files).

Remember to enter the last field or press <ctrl-Enter> to cause filtering to begin. After the Recursive Filtering function has completed, enter a Waveform Display/Edit menu with the FILENAME1 field set to “speechf1", the FILENAME2 field set to ”speech", the LOG AMP field disabled (set to ‘N’), the FORMAT field set to dual-trace (‘D’), and the FRAMESIZE field set to 128. You should see results similar to figure 10-2, below:

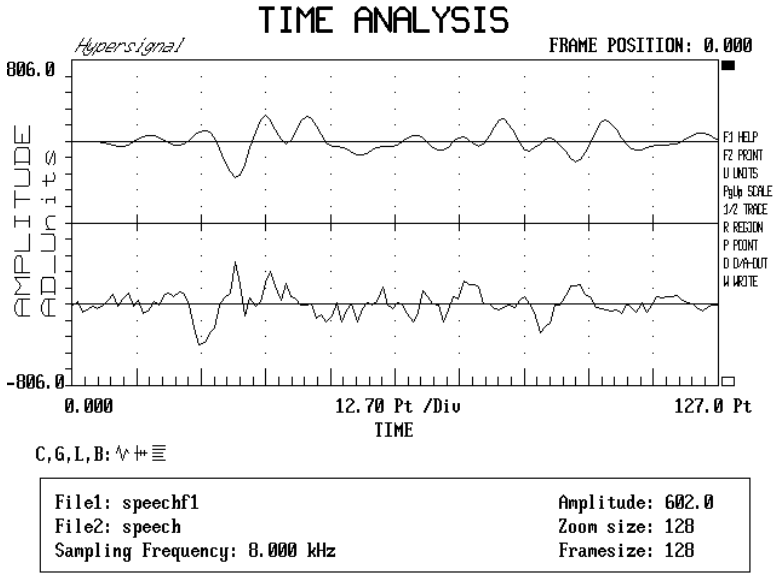


*Figure 10-2*

The relative size of the two waveforms should be about the same—this is because all Hypersignal IIR and FIR filters are designed to have unity gain. But, besides the obvious delay of the speechf1 waveform relative to the speech waveform, the speechf1 waveform has definitely been “smoothed”, or more accurately, lowpass-filtered. To measure the filtering effect more precisely, we can look in the frequency domain. Enter the following FFT Generation menu:

FILENAME: speech
FRAMESIZE: 128

| | |
|---|---|
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Hm |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields only apply if you are using a non-real-time software version, or you have the DSP ACCELERATOR field disabled ('none' entry) in the System Configuration menu. After the FFT Generation function finishes, return to the TIME DOMAIN column and enter another FFT Generation menu:

| | |
|---|---|
| FILENAME: | speechf1 |
| FRAMESIZE: | 128 |
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Hm |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

After this function finishes, enter the following 3-D Spectrogram menu:

| | |
|---|---|
| FILENAME1: | speechf1 |
| FILENAME2: | speech |
| POWER SPEC (P/M): | M |
| AMP SQUARE (Y/N): | N |
| LOG (N/+/-,N/base,Min): | N |
| SURFC, GRID(H/M/N,N/S/I): | H,N |
| Y-LABELS (SCALE,UNITS): | 1,Units |
| AMP SCALING (TOP,BOT,REF): | 0,0,A |
| ANGLE,TIME SHOWN (5-90, S): | 45,0 |

Press the 'Enter' key (or any button on the mouse) to begin waveform display. Now can you see just what data is missing from the "speechf1" waveform (on the top trace)?

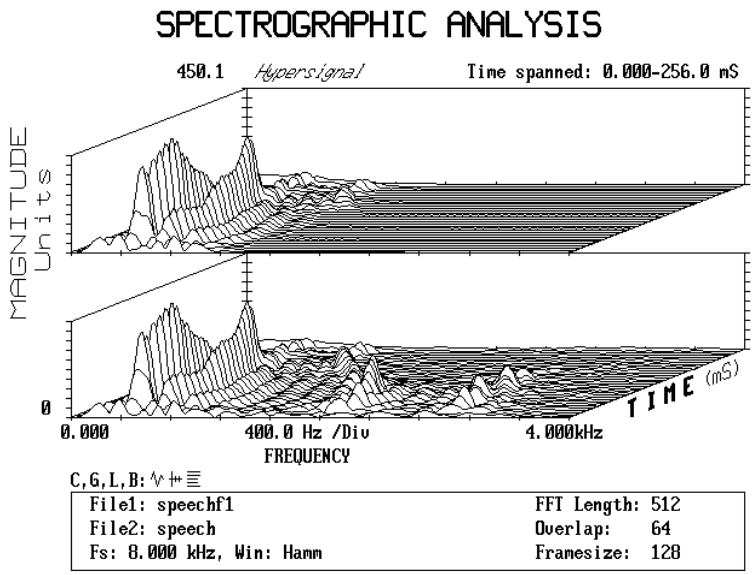


*Figure 10-3*

To make a final check, enter the following 3-D Spectrogram menu:

| | |
|---|---|
| FILENAME1: | speechf1 |
| FILENAME2: | butter2 |
| POWER SPEC (P/M): | M |
| AMP SQUARE (Y/N): | N |
| LOG (N/+/-,N/base,Min): | + |
| SURFC, GRID(H/M/N,N/S/I): | H,N |
| Y-LABELS (SCALE,UNITS): | 1,Units |
| AMP SCALING (TOP,BOT,REF): | 0,-75,A |
| ANGLE,TIME SHOWN (5-90, S): | 45,0 |

You should see results similar to figure 10-4, below:

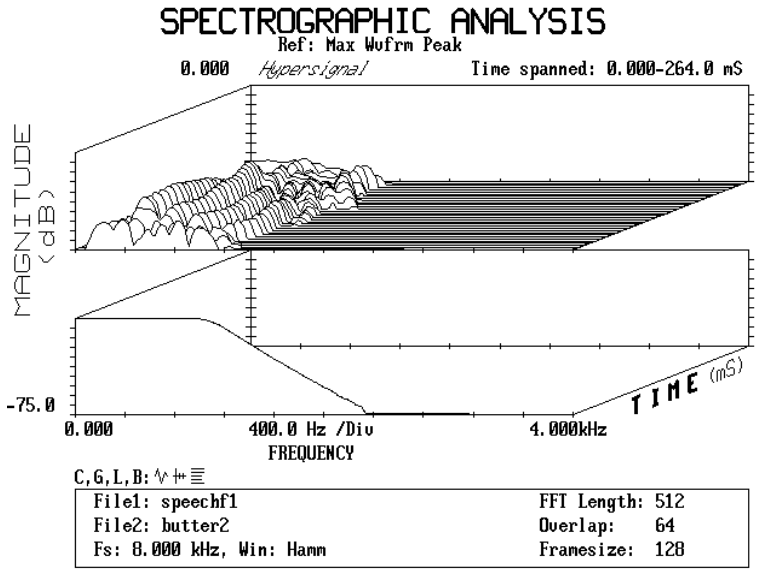


*Figure 10-4*

## 10.2 Convolution

Enter the following Convolution/Correlation menu to apply the “pm” FIR filter we designed earlier in section 9.4, Arbritary FIR Filters:

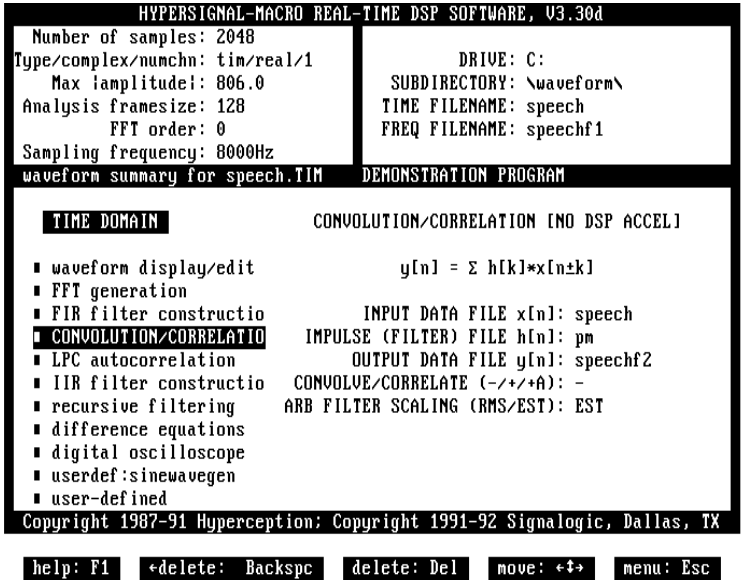


*Figure 10-5*

This example will take the “speech” waveform (should already be on your WAVEFORM subdirectory) and filter it with the “pm” FIR coefficient file, producing the “speechf2" waveform as output. Note that you could also directly use an FIR coefficient file of your own creation as the ”h[n]" file (see Users Manual section 4.8.3.1, Filter Coefficient File Format, for more information on creating your own filter coefficient files).

**POWER NOTE: Convolution can be used to apply both FIR *and* IIR filters to input waveforms. However, for IIR filters this sort of defeats the purpose, since applying the filter coefficients in recursive form (Recursive Filtering function, above) is much more efficient; the savings in computation time may be the main reason for using an IIR filter in the first place (see tradeoff discussion in chapter 9, DESIGNING FILTERS).**

After the Convolution/Correlation function has completed, enter a Waveform Display/Edit menu with the FILENAME1 field set to “speechf2", the FILENAME2 field set to ”speech", and the FRAMESIZE field set to 128. You should see results similar to figure 10-6, below:

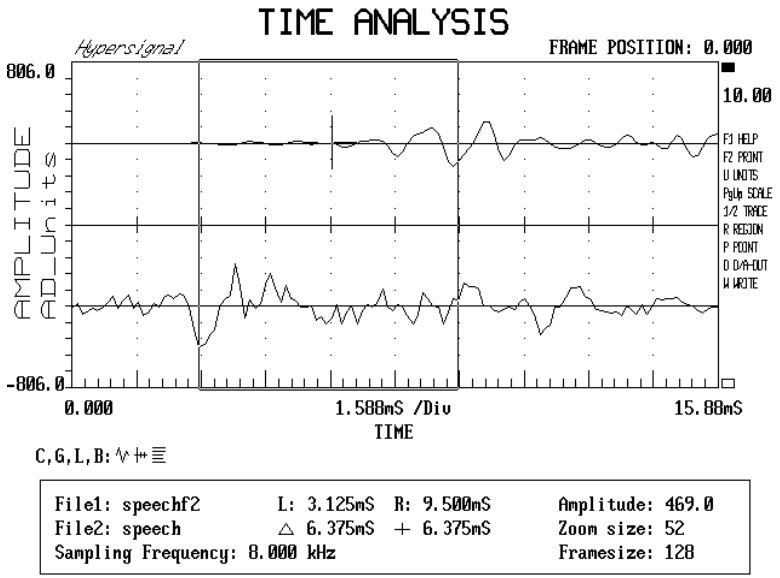


*Figure 10-6*

The relative size of the two waveforms should be about the same—this is because all Hypersignal IIR and FIR filters are designed to have unity gain. But the speechf2 waveform certainly shows the effect of some processing. To see exactly what, we can look in the frequency domain. Enter the following FFT Generation menu, and perform the FFT Generation:

FILENAME: speech
FRAMESIZE: 128
FFT ORDER: 9
% FRAMESIZE OVERLAP: 50
WINDOW (R/Hm/Hn/Bl/Bt): Hm
REAL/IMAG OUTPUT (Y/N): N
PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N

Remember that the PRECISION and PHASE/MAG APPROX fields only apply if you are using a non-real-time software version, or you have the DSP ACCELERATOR field disabled (‘none’ entry) in the System Configuration menu. After the FFT Generation function finishes, return to the TIME DOMAIN column and enter another FFT Generation menu:

FILENAME: speechf2
FRAMESIZE: 128

| | |
|---|---|
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Hm |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

After this function finishes, enter the following 3-D Spectrogram menu:

| | |
|---|---|
| FILENAME1: | speechf2 |
| FILENAME2: | speech |
| POWER SPEC (P/M): | M |
| AMP SQUARE (Y/N): | N |
| LOG (N/+/-,N/base,Min): | N |
| SURFC, GRID(H/M/N,N/S/I): | H,N |
| Y-LABELS (SCALE,UNITS): | 1,Units |
| AMP SCALING (TOP,BOT,REF): | 0,0,A |
| ANGLE,TIME SHOWN (5-90, S): | 45,0 |

Press the ‘Enter’ key (or any button on the mouse) to begin waveform display. Now can you see just what data is missing from the “speechf2" waveform (on the top trace)?

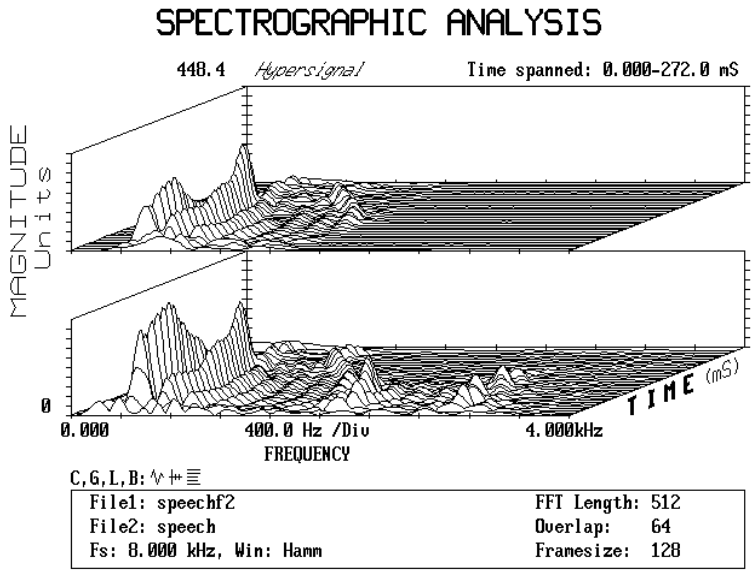


*Figure 10-7*

To make a final check, enter the following 3-D Spectrogram menu:

```
                FILENAME1: speechf2
                FILENAME2: pm
          POWER SPEC (P/M): M
         AMP SQUARE (Y/N): N
     LOG (N/+/-,N/base,Min): +
  SURFC, GRID(H/M/N,N/S/I): H,N
    Y-LABELS (SCALE,UNITS): 1,Units
AMP SCALING (TOP,BOT,REF): 0,0,A
ANGLE,TIME SHOWN (5-90, S): 45,0
```

Press the 'Enter' key (or any button on the mouse) to begin waveform display. Now can you see just what data is missing from the "speechf2" waveform (on the top trace)?

The display results of this menu are shown in figure 10-8, below. Note that the amplitude display format is linear magnitude. Try entering a '+' for the LOG field to show log magnitude format.

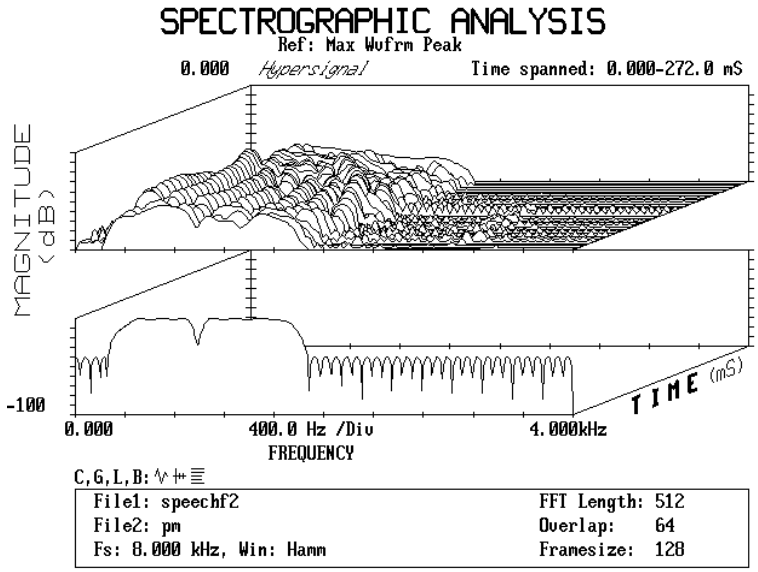


*Figure 10-8*

## 10.3 “Snap-In” Real-Time Filtering

When using Hypersignal-Workstation software, filters previously designed with the FIR and IIR Filter Construction function (and also user-generated filters created outside of Hypersignal) can be “snapped-in” to Hypersignal-Workstation real-time functions. For example, the Spectrum Analyzer function could show unfiltered data from channel 0 on trace 1 and filtered data on trace 2, or apply two different filters to channel 0 and channel 1 and show the results on trace 1 and 2. As another example, a filter could be applied in real-time to data being acquired/generated continuously to/from hard disk (either mono or stereo channel).

To apply the “chebyII” filter we designed earlier in section 9.1, Classical IIR Filters, in the Digital Oscilloscope function, enter the following Digital Oscilloscope menu:

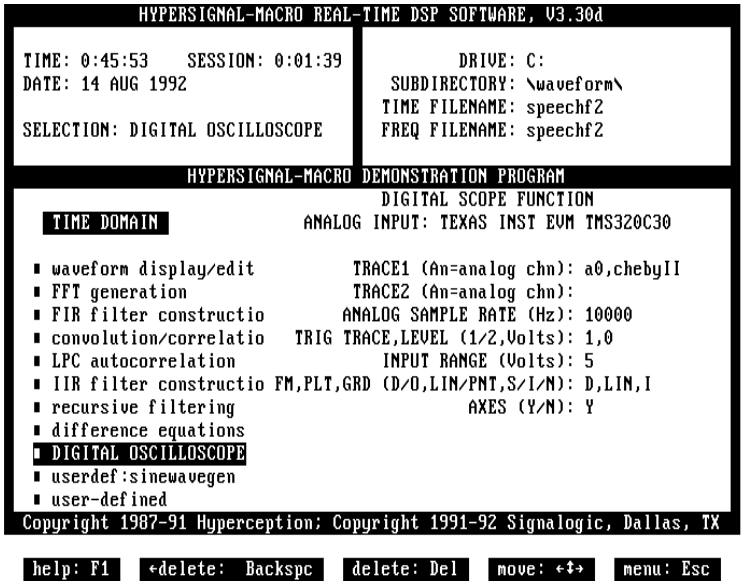


*Figure 10-9*

**NOTE: The “Analog Input” shown above should be set to the current DIGITAL OSCILLOSCOPE field in the “System Configuration” menu.**

This will apply the “chebyII” filter to incoming data on channel 0, and display the results on trace 1. We have purposefully kept the sampling rate low, in case the DSP chip on your currently installed board is one of the slower ones. To find out how fast your DSP chip can go and stay in real-time, slowly increase the SAMPLE RATE value until the display starts to show glitches in the data near the beginning of each frame or the display locks up altogether.

If you are using a fixed-point DSP device, you can control the filter quantization by entering the desired number of bits of quantization after the filter name. For example, a Digital Oscilloscope menu of

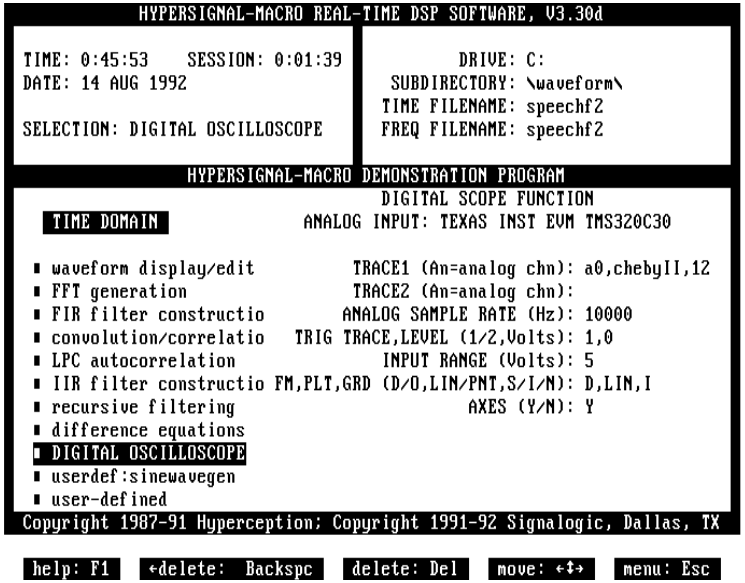


*Figure 10-10*

would reduce the quantization to 12 bits (coefficients ranging from -4096 to 4095). When no quantization is entered, the program will choose the highest value possible depending on filter structure and coefficients. You can also control the structure used for IIR filters by editing the structure field directly in the ASCII coefficient file (see Users Manual section

4.8.3.1, Filter Coefficient File Format, for details on the format and organization of filter coefficient files). Normally, the program will use an IIR cascade structure that is suitable for the type of filter being implemented.

To compare the “chebyII” filter’s frequency response file to its real-time frequency response in the Spectrum Analyzer function, first enter the following Spectrum Analyzer menu:

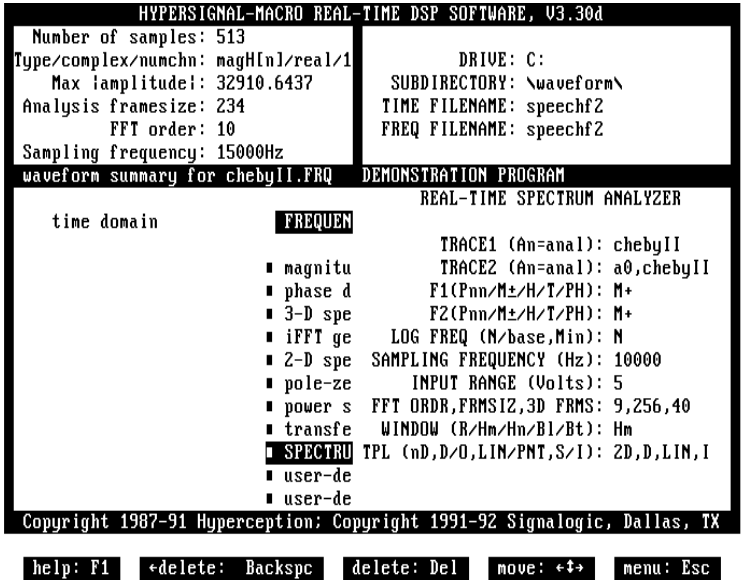


*Figure 10-11*

This menu will display the log magnitude of the “chebyII” frequency file on the top trace, and the log magnitude of live, filtered analog input from channel 0 on the bottom trace. By varying the frequency knob on your function generator, you should see the real-time frequency response track the outline of the file response. Use the ‘PgDn’ key to show more of the file response at one time vertically. The actual screen display would look similar to figure 10-12, below:

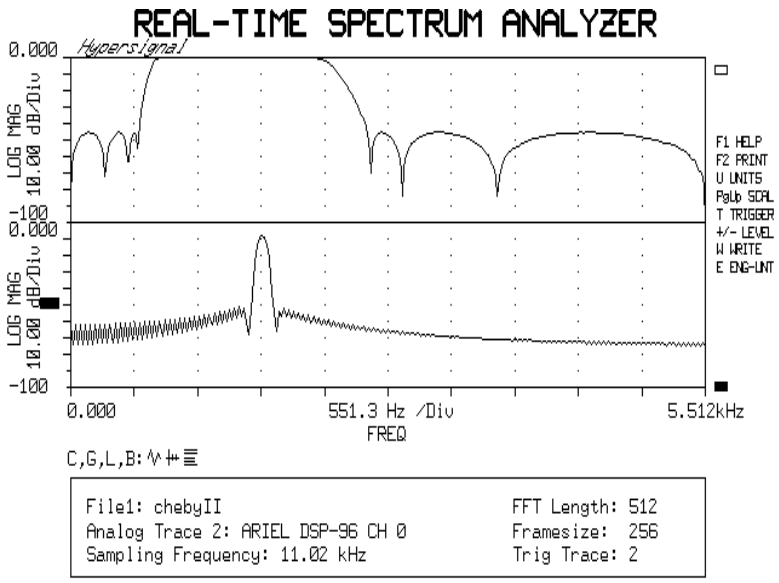


*Figure 10-12*

See chapter 11, SPECTRUM ANALYZER, for several detailed examples of Spectrum Analyzer operation. To apply filters to continuous disk data in real-time, use the Analog Conversion function. As an example, enter the following Analog Conversion menu:

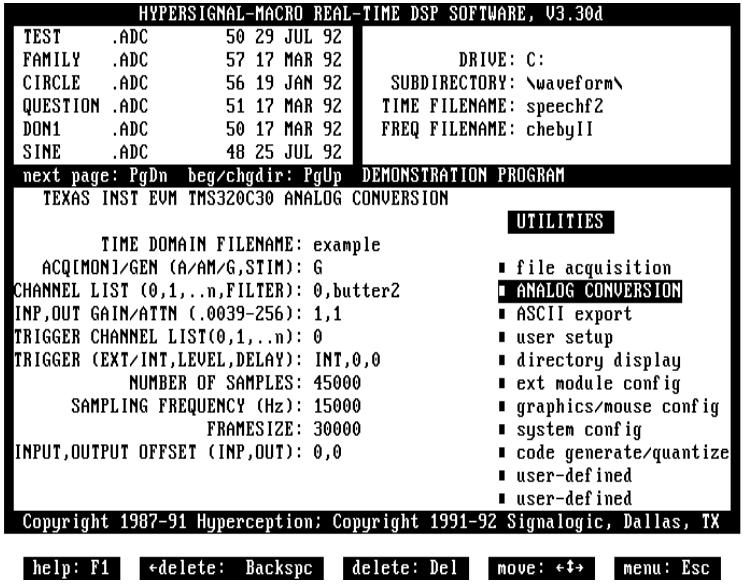


*Figure 10-13*

This menu will send the “example” file that we acquired in chapter 4, ACQUIRING DATA, to the D/A output while applying the “butter2" filter that we designed earlier in section 9.1, Classical IIR Filters, to the output data stream.

As another Analog Conversion example, consider the following menu that applies the “kaiser2" FIR filter we designed earlier in section 9.3, Classical FIR Filters, to a stereo recording:

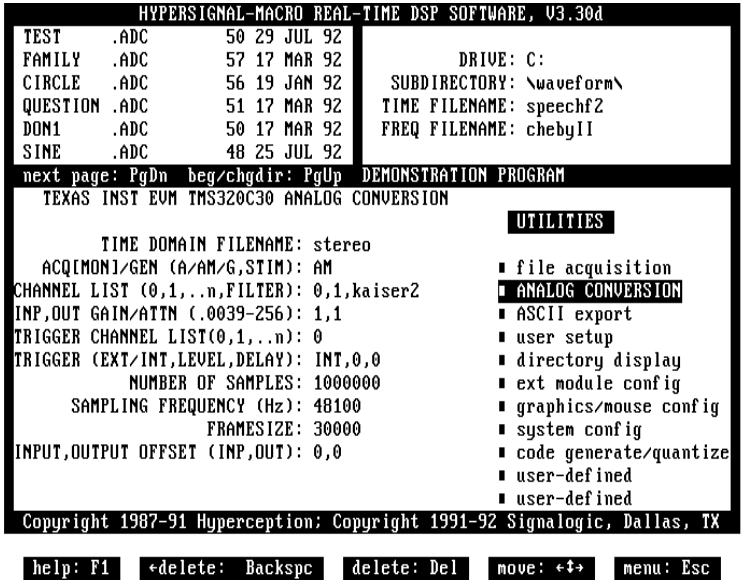


*Figure 10-14*

A few items of note about the above menu:

1) The CHANNEL LIST field has been set to 0,1 to record data on both channels 0 and 1.

2) The kaiser2 filter will be applied to both channels 0 and 1 simultaneously (the Analog Conversion function does not allow independent channel filtering, as do the Digital Oscilloscope and Spectrum Analyzer functions).

3) We are asking for 10000000 (10 million) samples at 48.1 kHz, or about 3.5 minutes of recording.

4) The ACQ/GEN field entry of ‘AM’ enables the “monitor” loopback mode, which digitally loops input around to output before and during the record. Since a filter is being applied, the loopback data is also filtered. Note that this implies that if you were to never answer the “BEGIN GENERATION (Y/N)” prompt, your system would act as an infinite, continuous real-time filter.

5) For a fixed-point DSP device, you can control the filter quantization by entering the number of bits of quantization after the filter name. For in-

stance, 14-bit quantization in the above example would require a CHANNEL LIST field entry of ‘0,1,kaiser2,14’.

**NOTE: There are several limitations to real-time filtering, including the DSP chip in use, the sampling rate, and the size of the filter. Taken together, these factors determine whether or not a given filter or filters can be executed in *real-time*. For example, newer DSPs with faster clock rates can run longer filters in real-time than older, slower DSPs. As another example, consider what happens when two independent filters are enabled simultaneously: all maximum parameters governing real-time operation are cut in half, because the DSP must work twice as hard. One potential problem that deserves your attention is maximum filter size (length, or number of taps for FIR filters, and number of cascade sections, or order for IIR filters). Two factors affect maximum size: the desired sampling rate, and the amount of DSP memory. The sampling rate problem is obvious: as the sampling rate increases, the maximum filter size decreases. As for the memory problem, DSPs are endowed with varying amounts of onchip RAM in which to hold filter coefficients and delay elements. Thus, even though the sampling rate may be within the realm of the possible, the amount of DSP onchip memory may become a limiting factor. In such cases, assuming a speed decrease is tolerable, it may be feasible to relocate coefficients and delay elements to external DSP memory by making direct DSP source code changes. For more information about real-time filtering limitations, see Users Manual section 4.9.3, Using “Snap-In” Real-Time Filtering.**

*this page intentionally left blank*

# 11.0 SPECTRUM ANALYZER

*The examples in this chapter require a real-time version of software.* The examples illustrate several measurements using the Spectrum Analyzer function, showing some of the many ways this instrument can be used. Data formats discussed include time, linear/log magnitude, phase, transfer function, and impulse response. Overall, the chapter is a good introduction to the Spectrum Analyzer instrument, which is a very in-depth software function within Hypersignal, and could be a separate program by itself.

The examples in this chapter assume that a function generator producing a periodic signal (sine wave, square wave, etc.) is connected to channel 0 of the currently installed DSP/Analog hardware. It is further assumed that you have already designated the hardware in the SPECTRUM ANALYZER field of the System Configuration menu. For more information about system configuration related to plug-in DSP/Analog hardware, see section 2.3, Configuring Hypersignal.

**NOTES: If you are using an analog-only board (no DSP chip) for your analog input, you must have a "DSP Coprocessor" board installed as your DSP ACCELERATOR (see the System Configuration discussion in section 2.3, Configuring Hypersignal).**

**Much of the information in this chapter also applies to operation of the Digital Oscilloscope instrument, especially the following sections: 11.1, Sources and Combinations of Input, 11.1.1, Real-Time Filtering of Analog Input, 11.1.2, Dual- and Overlaid-Trace Display, and 11.6, Timebase, Triggering, Vertical Scaling & Position, and Other Controls.**

## 11.1 Sources and Combinations of Input

Input sources for the Spectrum Analyzer and Digital Oscilloscope functions can be analog input channels carrying data from the currently in-

stalled DSP/Analog hardware or standard Hypersignal waveform files. Since the sources are varied, reference is made throughout the Hypersignal documentation to “traces”, implying a physical-to-trace mapping, where data is coming from physical sources and being updated continuously in one or two of two possible display traces. Data can be displayed as single-trace (display of trace 1), dual-trace (half-area display of both trace 1 and trace 2), or overlaid-trace (full-area, overlaid display of both trace 1 and trace 2).

Mixed display of analog input and waveform file input is possible; analog display from different channels (referred to as “interchannel”) or from the same channel (“intrachannel”). The following combinations of input sources are allowed:

one analog trace
two analog traces
one file trace
one file trace and one analog trace
two file traces

**NOTE: When displaying both analog and waveform file input, the waveform file input must be specified for trace 1 display.**

Like intrachannel analog input, dual-display from file input can refer to the same file, but this type of operation is not often used. However, intrachannel analog input can be especially useful for viewing the same analog data simultaneously in different formats (e.g. time domain and frequency domain), or when applying real-time digital filters (filtered data vs. unfiltered data).

Specification of the source(s) of input is made in the TRACE1 and TRACE2 fields. Analog channels are specified as ‘aN’, where N is the physical channel number (usually starting with zero, but depends the current on hardware—see board-specific sections in the Hardware Reference Guide). Below are some abbreviated menu examples of the combinations listed above:

<u>one analog trace</u>

TRACE1: a0
TRACE2:

two analog traces

TRACE1: a0
TRACE2: a1

two analog traces, cont.

TRACE1: a0
TRACE2: a0

two analog traces, cont.

TRACE1: a2
TRACE2: a4

one file trace

TRACE1: signal
TRACE2:

one file trace and one analog trace

TRACE1: signal
TRACE2: a0

one file trace and one analog trace, cont.

TRACE1: signal
TRACE2: a1

two file traces

TRACE1: signal1
TRACE2: signal2

Remember to use the 'Delete' or 'BackSpace' keys to blank a field, if necessary. In the Spectrum Analyzer, specification of the format in which to display data is made in the F1 and F2 fields (see sections 11.2 through 11.4 below). For the Digital Oscilloscope function, the Fn fields do not exist, since all data is assumed to be in a time domain format.

## 11.1.1 Real-Time Filtering of Analog Input

Digital FIR and IIR filters already designed using the Filter Construction functions in Hypersignal, or input directly to an ASCII coefficient file from user-defined filters, can be applied in real-time to incoming analog data in the Spectrum Analyzer and Digital Oscilloscope instruments.

**NOTE: There are many limitations concerning real-time filtering, which usually depend on the DSP chip being used, the sampling rate, and the length, or size, of the filter. See section 10.3, "Snap-In" Real-Time Filtering, for further explanation of these potential problems, and Users Manual section 4.9.3, Using "Snap-In" Real-Time Filtering, for fully detailed information.**

Real-time filters can be specified on an independent analog channel basis in the TRACE1 and TRACE2 fields. Below are some partial menu examples of filter specification:

trace 1 displaying channel 0 data as filtered, trace 2 displaying channel 0 data as unfiltered

TRACE1: a0,elliptic
TRACE2: a0

trace 1 displaying channel 0 data as filtered, trace 2 displaying channel 1 data as filtered with the same filter

TRACE1: a0,kaiser
TRACE2: a1,kaiser

trace 1 displaying channel 0 data as filtered, trace 2 displaying channel 1 data as filtered with a different filter

TRACE1: a0,kaiser
TRACE2: a1,cheby2

trace 1 displaying the filter frequency response (from a waveform file), and trace 2 displaying channel 0 data as filtered

TRACE1: elliptic
TRACE2: a0,elliptic

Normally, specifying an IIR or FIR filter is simply a question of the filename, because the built-in Hypersignal Filter Construction functions will not allow FIR and IIR filters to use the same waveform filenames. When IIR filters are specified, the program automatically reads the appropriate cascade structure from the coefficient file, and applies the filter accordingly. When a fixed-point DSP chip is in use, the program will choose the maximum possible quantization for the filter coefficients. However, you can also specify quantization explicitly, as shown in the partial menu example below:

TRACE1: a0,kaiser,12

TRACE2: a1,cheby2

In the above example, 12-bit coefficient and delay element quantization has been specified, and will apply to *both filters*, since the program does not allow independent quantization values.

**POWER NOTE: Filter coefficients and IIR cascade structure are read directly from ASCII "filter coefficient" files, which are normally produced by the built-in Hypersignal Filter Construction functions. Because of the ASCII, documented nature of the coefficient file format, it is possible to enter user-defined coefficients directly into these files; the format even supports factored or direct IIR forms. Since these files serve as input for all post-design filter-related functions and displays in Hypersignal, including Code Generation, user-defined filters can be used in place of or side-by-side with Hypersignal-designed filters. See Users Manual section 4.8.3.1, Filter Coefficient File Format, for more information.**

## 11.1.2 Dual- and Overlaid-Trace Display

When both the TRACE1 and TRACE2 fields are not blank, then "dual-trace operation" will occur. Although the software, and possibly the DSP chip, is processing two traces, there are different display possibilities. The most commonly used is "dual-trace", which splits the display area into two display areas vertically, displaying trace 1 at the top and trace 2 underneath. In the Spectrum Analyzer, this mode of operation is specified by entering a 'D' in the FORMAT subfield of the TOPOLOGY (TPL) field, and in the FORMAT (FM) subfield for the Digital Oscilloscope. Below is a partial data-entry menu example of dual-trace entry for the Spectrum Analyzer:

TPL (nD,D/O,LIN/PNT,S/I): 2D,D,LIN,I

Below is a partial data-entry menu example of dual-trace entry for the Digital Oscilloscope:

FM,PLT,GRD (D/O,L/P,S/I/N): D,LIN,I

Once inside the instrumentation display, there are two small rectangles located at the right edge of the display area that indicate the "current active trace", for which the instrument controls apply (see section 11.6,

Timebase, Triggering, Vertical Scaling & Position, and Other Controls, below). A filled-in, or solid, rectangle indicates the current active trace. Also, *for time data format display only*, a trigger level indicator will appear at the lefthand edge of the current active trace display area (pressing the '+' and '-' keys moves the level indicator up and down). Pressing the '1' and '2' keys inside the display will toggle the current active trace. It is also possible to invoke "lock-step" operation, where controls apply to both traces simultaneously. This is done by pressing a 'Shift-1' or 'Shift-2' key sequence, after which *both* trace indicators will become solid. Pressing the the '1' or '2' keys always deactivates lock-step operation.

In addition to dual-trace display, it is possible to select an overlaid-trace mode of display, where both trace 1 and trace 2 occupy the entire display area. In the Spectrum Analyzer, this mode of operation is specified by entering an 'O' in the FORMAT subfield of the TOPOLOGY field, and in the FORMAT subfield for the Digital Oscilloscope. Below is a partial data-entry menu example of overlaid-trace entry for the Spectrum Analyzer:

TPL (nD,D/O,LIN/PNT,S/I): 2D,O,LIN,I

Below is a partial data-entry menu example of overlaid-trace entry for the Digital Oscilloscope:

FM,PLT,GRD (D/O,L/P,S/I/N): O,LIN,I

In overlaid-trace display, the current active trace indicator and trace selector keys work as in dual-trace display. Another possible form of display operation is dual-trace *operation*, but single-trace *display*. In the Spectrum Analyzer, this mode of operation is specified by entering a '1' or '2' in the FORMAT subfield of the TOPOLOGY field, and in the FORMAT subfield for the Digital Oscilloscope, depending on which trace you would like to see. Below is a partial data-entry menu example of single-trace entry for the Spectrum Analyzer:

TPL (nD,D/O,LIN/PNT,S/I): 2D,1,LIN,I

Below is a partial data-entry menu example of single-trace entry for the Digital Oscilloscope:

FM,PLT,GRD (D/O,L/P,S/I/N): 1,LIN,I

In the Spectrum Analyzer only, it is also possible to enable single-trace display during dual-trace operation by using an 'N' entry in, or blanking out, either the F1 or F2 fields (see sections 11.2 through 11.4, below) of

the data-entry menu, depending on which trace should be disabled. Finally, it is important to remember that single-trace entry should only be used for dual-trace operation, in which case both the TRACE1 and TRACE2 fields are not blank. The FORMAT subfields are ignored for single-trace operation, where the TRACE2 field is blank.

## 11.2 Time Measurement

In the Digital Oscilloscope, data is always displayed in a time domain format, so no data format fields are present on the data-entry menu. However, in the Spectrum Analyzer, many different data formats are possible. These are chosen using the F1 and F2 (FORMAT1 and FORMAT2) fields, which specify the format of data coming from the input sources indicated by the TRACE1 and TRACE2 fields, respectively.

A 'T' entry in the Spectrum Analyzer F1 or F2 fields selects time format, which is a time domain format, for the data indicated by the corresponding TRACEn field. Time format is only valid for analog input; waveform file input in the corresponding TRACEn field will produce an error message. Below are some partial menu examples of time format specification:

trace 1 displaying channel 0 data in time format, trace 2 deactivated

TRACE1: a0
TRACE2:
F1: T
F2:

In the above example, the F2 field is ignored, since the TRACE2 field is blank. Also, entry after the 'T' is generally disregarded, because such "suffix" entry is commonly used for other data formats as explained below, allowing formats to be switched easily by simply changing the first letter of the entry.

trace 1 displaying channel 0 data in time format, trace 2 displaying channel 1 data also in time format

TRACE1: a0
TRACE2: a1
F1: T
F2: T

In the Digital Oscilloscope, the amount of time data displayed is initialized by the software to a little under half the available memory capacity of the DSP board or the Digital Oscilloscope internal memory, whichever is

smaller. Thereafter, the amount of time data displayed is controlled inside the Digital Oscilloscope display by the timebase controls, which are the left and right arrows or left and right mouse movement (decrease, increase the amount of data, respectively). In the Spectrum Analyzer, the amount of time data displayed is controlled by the FRAMESIZE field, which is also one of the fundamental Fourier analysis parameters, along with FFT size and time domain window. These Fourier parameters, along with several F1 and F2 field frequency domain format entries, are discussed in section 11.3, Frequency Measurement, below.

Following is an actual data-entry menu and Spectrum Analyzer display example demonstrating time measurement. For this example, *it is assumed that you have a function generator connected to channel 0 of your DSP/Analog board, and generating a periodic signal.* For the tutorial examples in this section, a sine wave was used. Enter the Spectrum Analyzer menu below:

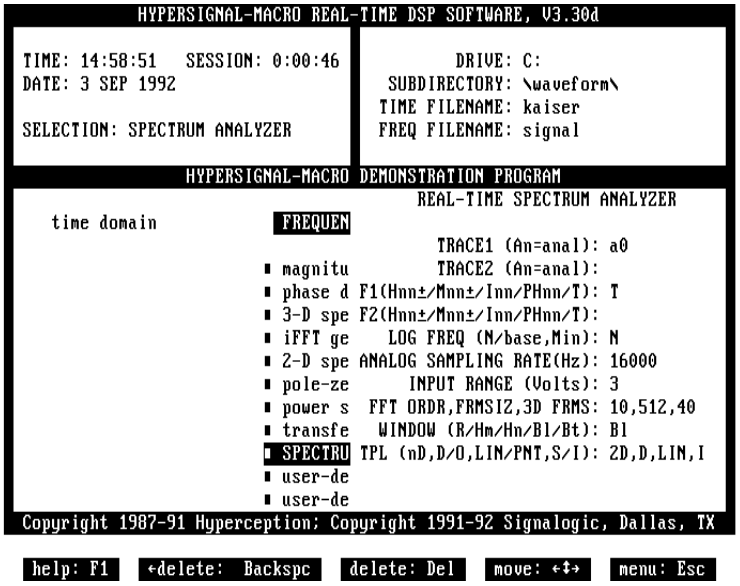


*Figure 11-1*

The menu above directs channel 0 analog input data to be displayed in time format on trace 1. Trace 2 is blank, or inactive; therefore trace 1 will occupy the whole display area. Note that the setting of the LOG FREQ field, which is explained in section 11.3, Frequency Measurement, below,

is ignored for traces displaying time format data, since this field only controls traces containing a frequency axis. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-2, below:

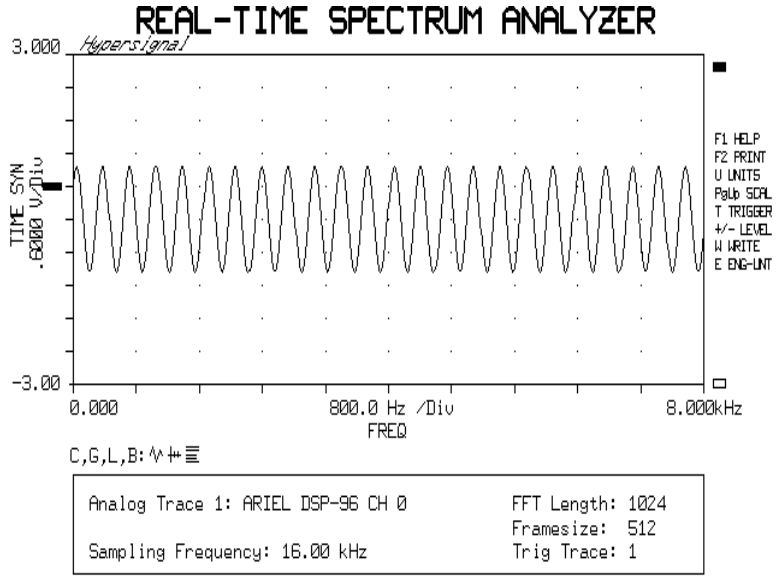


*Figure 11-2*

Note that in the above display, the trigger level indicator (small rectangle at lefthand edge of display area) has been increased from zero to nearly the maximum amplitude of the time data, causing displayed data to begin with a rising slope of this level. Negative trigger levels will cause a falling slope, and negative levels. Trigger levels within a very small hysteresis area about zero (the default value) direct the program to operate in "free-run" mode, where any input value begins display of the next frame of data. The trigger level is controlled with the '+' and '-' keys; in the example above the '+' key was pressed several times to bring the trigger level up near the maximum amplitude of the sine wave. The sampling frequency was 16 kHz, yielding a bandwidth of about 8 kHz (depending on the anti-aliasing filters used by the DSP/Analog hardware), and the frequency of the sine wave input is around 750 Hz. Note that the frequency of the sine wave could be measured by taking the inverse of 1 period of the sine wave, but a much easier method is to display frequency directly, as explained in the following sections.

## 11.3 Frequency Measurement

This section only applies to the Spectrum Analyzer, which, through complex communication and utilization of one of the supported DSP/Analog boards or systems, can produce a continuously updating frequency domain display. Fast Fourier Transform (FFT) analysis is performed in real-time on incoming analog data to provide basic input data. Various parameters can be entered in the data-entry menu to control the Fourier analysis, including time domain framesize, time domain window, and FFT length. Magnitude, phase, transfer function, and impulse response formats are possible. (The impulse response format is a time domain format, but is derived from transfer function calculation, and thus still depends on various Fourier parameters, which are normally associated with frequency domain formats.)

The partial Spectrum Analyzer data-entry menu example below would enable a 1024 pt. FFT length, a 512 pt. time framesize, and a Hamming window:

trace 1 displaying channel 0 data as linear magnitude, trace 2 also displaying channel 0 data, but in time format

```
                    TRACE1: a0
                    TRACE2: a0
                        F1: M
                        F2: T
FFT ORDR,FRMSIZ,3D FRMS: 10,512,40
   WINDOW (R/Hm/Hn/Bl/Bt): Hm
```

The ‘M’ entry above specifies a linear magnitude format, which is a frequency domain data format. Magnitude is the basic amplitude result of the Fourier analysis, showing amplitude, or energy, versus frequency. Other frequency domain formats, which are illustrated in subsequent data-entry menu examples, include phase and transfer function. The plain ‘M’, with no ‘+’ or ‘-’ “log” suffix (see examples below) specifies linear magnitude format. Note that the FFT length is actually specified as the “FFT order”, or as a power of 2, or in this case $2^{10} = 1024$. The time domain framesize, in this case 512, specifies the amount of analog data that is acquired before FFT processing—processing which is performed continuously and in real-time by the DSP chip, frame after frame. Note that when the time domain framesize is less than the FFT length, “zero-filling” of the FFT input, after the end of the framesize, takes place. Zero-filling can be used to increase the frequency resolution of a desired time framesize chosen because of the nature of the data. See Users Manual sec-

tion 4.4.2, Zero-Filling and Truncation Discussion, for more information. The 3D FRMS field entry has to do with the "3-D Map Mode" of Spectrum Analyzer operation, of which example data-entry menus and displays are shown in section 11.8, 3-D Map Mode Display. The 'Hm' entry in the WINDOW field specifies a Hamming window. Other possibilities include an 'R' entry for a rectangular window (i.e. no window), and a 'Bl' entry for a Blackman window, which tapers smoothly to zero at both ends. Windows are applied symmetrically with respect to time framesize (not FFT length), and are often used to enhance the signal to noise ratio of the resulting frequency domain display, at the expense of "smearing", or widening the true signal frequencies. Essentially, windows work by eliminating the "edge noise" of non-zero time domain amplitude energy occuring at the beginning and end of the time data frame being input to the FFT. Otherwise, amplitude energy occuring at the ends of the framesize must by definition be abrupt, and thus carry wideband spectral content, producing artifacts in the resulting frequency display.

**NOTES: The FFT ORDER and FRAMESIZE and fields do not apply to waveform file input (see below). If neither TRACEn field specifies analog input, then these two fields will be completely ignored.**

**The maximum time domain framesize in the Spectrum Analyzer is equal to the maximum FFT length. The maximum FFT length will vary between DSP boards, depending on the programming of the DSP chip and the amount of memory available on the board. For example, some boards may not have enough memory to support the maximum FFT length of which the DSP chip is capable for two-trace analog input operation. In such a case, to achieve two-trace analog input operation, you would have to decrease the FFT length.**

Below is a partial data-entry menu example similar to the previous one, but specifying a log magnitude format:

trace 1 displaying channel 0 data as log magnitude, trace 2 also displaying channel 0 data, but in time format

TRACE1: a0
TRACE2: a0
F1: M+

F2: T

Note the use of the '+' suffix to indicate log magnitude, instead of linear magnitude. A '-' suffix would indicate negative log magnitude, with the trace display "flipped", or rotated.

Following is an actual data-entry menu and Spectrum Analyzer display example demonstrating frequency measurement. For this example, *it is assumed that you have a function generator connected to channel 0 of your DSP/Analog board, and generating a periodic signal.* For the tutorial examples in this section, a sine wave was used. Enter the Spectrum Analyzer menu below:

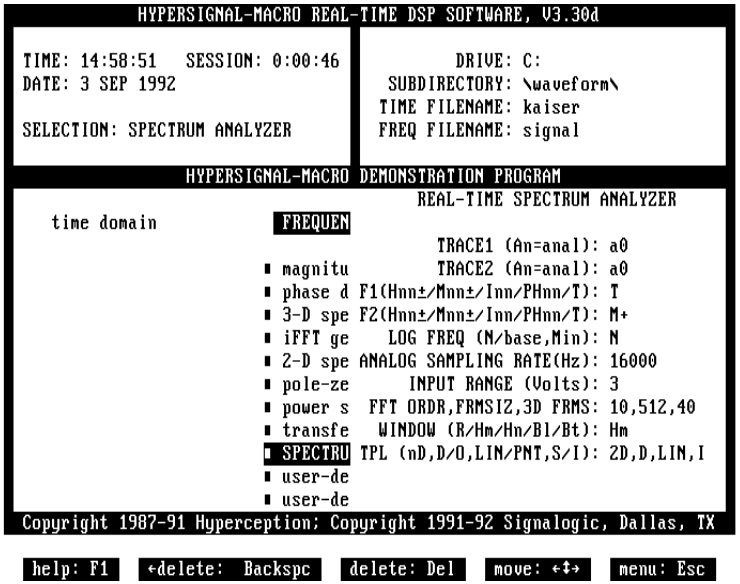


*Figure 11-3*

As with Figure 11-1, the menu above directs channel 0 analog input data to be displayed in time format on trace 1. However, unlike Figure 11-1, trace 2 has been activated to contain log magnitude data, also from channel 0 analog input. Note that the setting of the LOG FREQ field, which is explained below, is no longer ignored and now applies to the trace 2 display, as it contains a frequency axis. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-4, below:

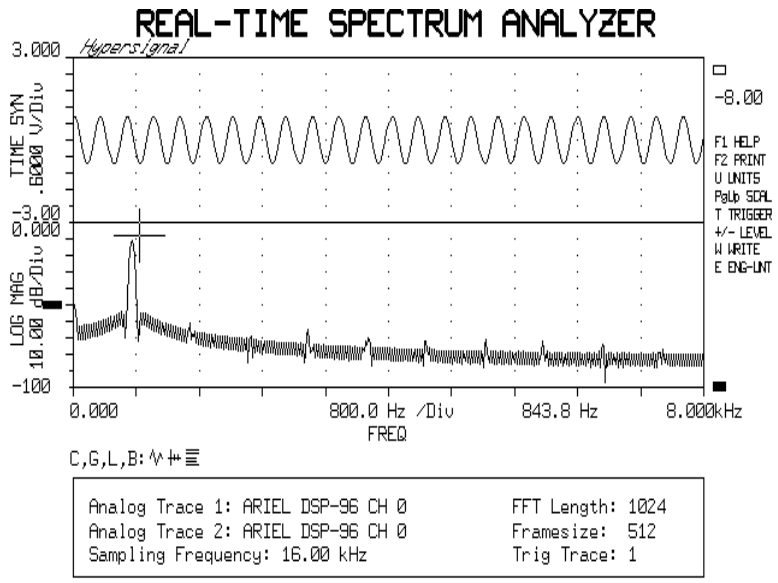


*Figure 11-4*

**NOTE: For boards with a 16-bit fixed-point DSP device you will only be able to see log magnitude precision to about -67 dB. This is because Hypersignal currently does only a single-precision FFT on these devices.**

In a dual-trace display such as the one shown above, pressing the '1' and '2' keys will toggle the "current active trace" between trace 1 and trace 2. Most controls, including trigger level, scaling, crosshair cursor, etc. apply to the current active trace. The current active trace is indicated by two small rectangles at the righthand edge of the display area, with the active trace rectangle appearing solid, or filled-in. Trigger level control and trigger operation are active as explained after Figure 11-2, above. In this case, since both traces are displaying data from the same analog channel, setting the trigger level indicator for either trace will affect the data in both traces. The sampling frequency was 16 kHz, yielding a bandwidth of about 8 kHz (depending on the anti-aliasing filters used by the DSP/Analog hardware), and the frequency of the sine wave input is around 750 Hz, which is easily read from trace 2. Try pressing the '2' key to select trace 2 as the current

active trace, and then moving the mouse so that the crosshair cursor is located near the frequency peak. Note the amplitude readout at the righthand side of the display, and the frequency readout along the horizontal axis, both of which update in response to crosshair cursor movement.

**NOTE: The following is a very important note, and might save you some frustrating moments. Even though the trigger level indicator, and the effect of its controls ('+' and '-' keys), change with the current active trace ('1' and '2' keys), the actual *trigger trace* only changes with the 'T' key, which toggles the trigger trace between trace 1 and trace 2. There are various reasons for this, including:**

**1) single-shot operation for a trace is allowed by setting the trigger level, and then switching the trigger to that trace**

**2) the trigger level indicator can be used to mark time domain format amplitude levels**

**3) the trigger level indicator serves as an additional, convenient indicator of the current active trace**

**Note that in the Digital Oscilloscope function, the initial trigger trace can be set from the data-entry menu.**

In place of or in combination with analog input, the Spectrum Analyzer and Digital Oscilloscope functions can also accept waveform file input. Waveform files can be any of the following:

1) generated by any other Hypersignal function, including filter design, difference equations, FFT Generation, etc.

2) captured inside the Spectrum Analyzer or Digital Oscilloscope functions and now being displayed simultaneously with analog input for comparison purposes

3) imported by the File Acquisition or ASCII-to-Binary Conversion functions from another program or external source

For more information about waveform files, see Users Manual section 4.1, Getting Waveforms In and Out of Hypersignal. Below is a partial

data-entry menu example similar to the one above, but indicating waveform file input:

trace 1 displaying waveform file data as log magnitude, trace 2 displaying channel 0 data as log magnitude

TRACE1: signal
TRACE2: a0
F1: M+
F2: M+

Waveform file input can be shown in linear, log, or negative log formats. If a power spectra version of the waveform exists, it can also be shown in these formats, but averaging must be enabled (see below). Following is a partial data-entry menu example similar to the one above, but additionally specifying a phase format:

trace 1 displaying channel 0 data as log magnitude, trace 2 also displaying channel 0 data, but in phase format

TRACE1: a0
TRACE2: a0
F1: M+
F2: PH

**NOTES: The ANALOG SAMPLING RATE, INPUT RANGE, FFT ORDER, and FRAMESIZE and fields do not apply to waveform file input. If neither TRACEn field specifies analog input, then these fields will be completely ignored.**

**In the Spectrum Analyzer function, waveform file input can only be displayed in linear magnitude, log magnitude, and transfer function formats, or averaged (power spectra) versions of these formats. Attempting to display waveform file data in time, phase, or impulse response formats will result in an error message.**

Following is an actual data-entry menu and Spectrum Analyzer display example demonstrating frequency measurement of mixed analog and waveform file input. For this example, *it is assumed that you have a function generator connected to channel 0 of your DSP/Analog board, and generating a periodic signal.* For the tutorial examples in this section, a sine wave was used. Enter the Spectrum Analyzer menu below:

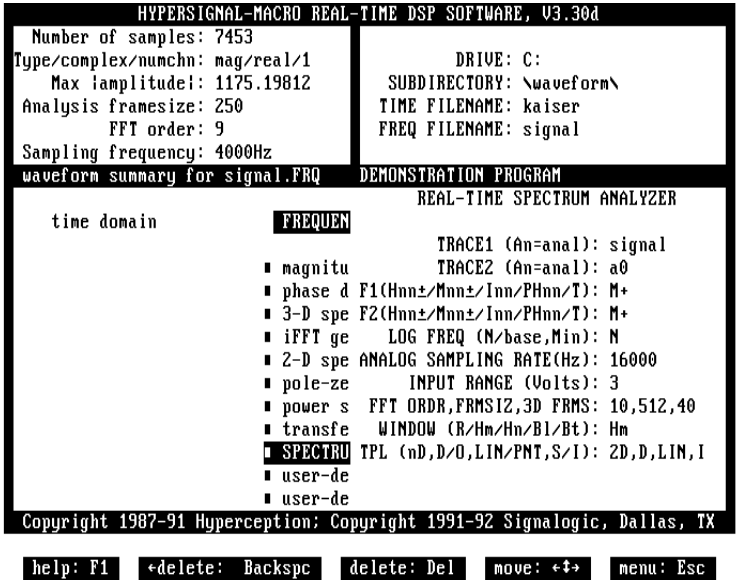


*Figure 11-5*

The menu above directs input from the frequency domain waveform file "signal" (provided on distribution diskette #3) to be displayed on trace 1 in log magnitude format, and analog input from channel 0 to be displayed on trace 2, also in log magnitude format. Note that the setting of the LOG FREQ field, which is explained below, affects the data displayed in both traces, because both contain a frequency axis. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-6, below:

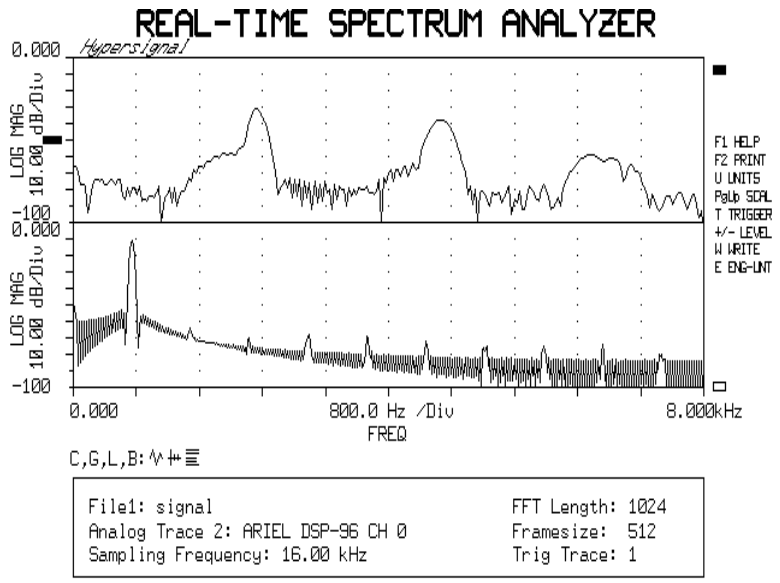


*Figure 11-6*

Note that the waveform file input display repeats, as it reaches either the end of the waveform of the maximum amount of memory available to contain it. As explained after Figure 11-4 above, pressing the '1' and '2' keys will select the current active trace. All controls work in the same way for waveform file and analog input except for triggering, which is not active in the Spectrum Analyzer for waveform file input (but is in the Digital Oscilloscope; see section 11.6 below for an explanation). Note also that the two traces could be overlaid for comparison purposes, a frequent technique when working with previously stored (file) input, by entering an 'O' in the FORMAT subfield on the data-entry menu. See section 11.1.2, Dual- and Overlaid-Trace Display for example of how to do this.

Averaging of frequency domain formats, which can produce much more stable and "measurable" signals, is also possible. Averaging is specified by entering a number of frames to average as a suffix to the format entry. For example, below is a partial data-entry menu example similar to previous examples, but specifying an averaged linear magnitude format and an averaged log magnitude format simultaneously:

trace 1 displaying channel 0 data as averaged log magnitude, trace 2 displaying channel 1 data as averaged linear magnitude

```
TRACE1: a0
TRACE2: a1
    F1: M20+
    F2: M50
```

In the above example, 20 frames are averaged on trace 1, and 50 on trace 2. Waveform file input can also be shown in be shown in averaged linear, log, and negative log formats, if a power spectra version of the waveform exists. Power spectra versions of frequency domain waveform files are created by the Power Spectra Generation function. Following is a partial data-entry menu example similar to the one above, but specifying an averaged waveform file entry for trace 2:

trace 1 displaying waveform file data as averaged log magnitude, trace 2 displaying channel 1 data as averaged linear magnitude

```
TRACE1: signal
TRACE2: a1
    F1: M20+
    F2: M50
```

In this case the '20' frames entry in the F1 field is ignored, other than to specify the power spectra version of the waveform file, because averaging of the waveform has already taken place, according to menu data-entry in the Power Spectra Generation function. Also, in order to be consistent with other, file-based frequency domain displays, a 'P' (power spectra) entry can be used to specify averaging in the Spectrum Analyzer F1 and F2 fields. The example below produces identical functional results to the previous example:

```
TRACE1: signal
TRACE2: a1
    F1: P20+
    F2: P50
```

Note that averaging can easily be turned on and off by simply making the number of frames equal to zero, conveniently allowing the format and log entries to remain in position:

```
TRACE1: signal
TRACE2: a1
    F1: M00+
    F2: M00
```

In the above example, no averaging will occur. Following is an actual data-entry menu and Spectrum Analyzer display example of averaged frequency format display. For this example, *it is assumed that you have a*

*function generator connected to channel 0 of your DSP/Analog board, and generating a periodic signal.* For the tutorial examples in this section, a sine wave was used. Enter the Spectrum Analyzer menu below:

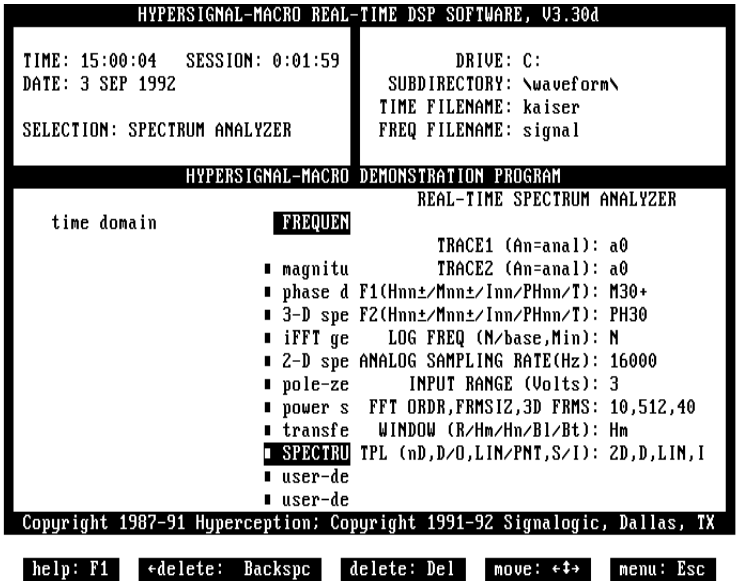


*Figure 11-7*

The menu above directs input from analog channel 0 to be displayed on trace 1 in averaged log magnitude format, and analog input from channel 0 to be displayed on trace 2 in averaged phase format. Note that the setting of the LOG FREQ field, which is explained below, affects the data displayed in both traces, because both contain a frequency axis. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-8, below:

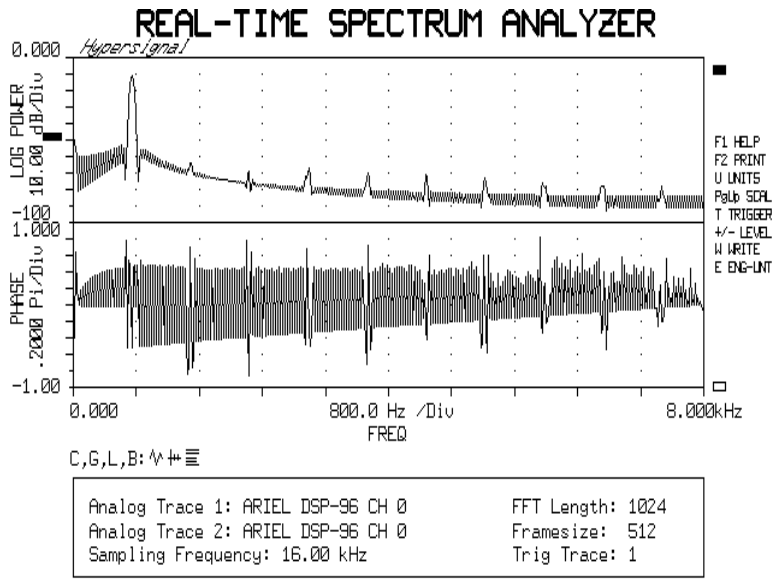


*Figure 11-8*

In the display above, the trigger level has been increased somewhat to produce a steadier, more “measurable” display. This is highly recommended for phase displays, because phase response is especially sensitive to non-triggered time data, which causes random phase shifts in the time data from one frame to the next. As explained after Figure 11-4 above, pressing the ‘1’ and ‘2’ keys will select the current active trace. As explained after Figure 11-2, above, the ‘+’ and ‘-’ keys control trigger level (even though you can’t see the time data)—remember that the ‘T’ key selects the actual trigger trace, independent of the current active trace (see second NOTE after Figure 11-4, above).

Once inside the display, press the ‘1’ key to select trace 1, and then press the ‘W’ key to invoke the write-to-file, or capture, command. If you have Hypersignal-Acoustic software, ignore the various other types of data output prompts—all we want is the basic power spectra (averaged) log magnitude output. Use an entry of ‘logavg’ to answer the filename prompt. Now return to the data-entry menu and enter the following Spectrum Analzyer menu:

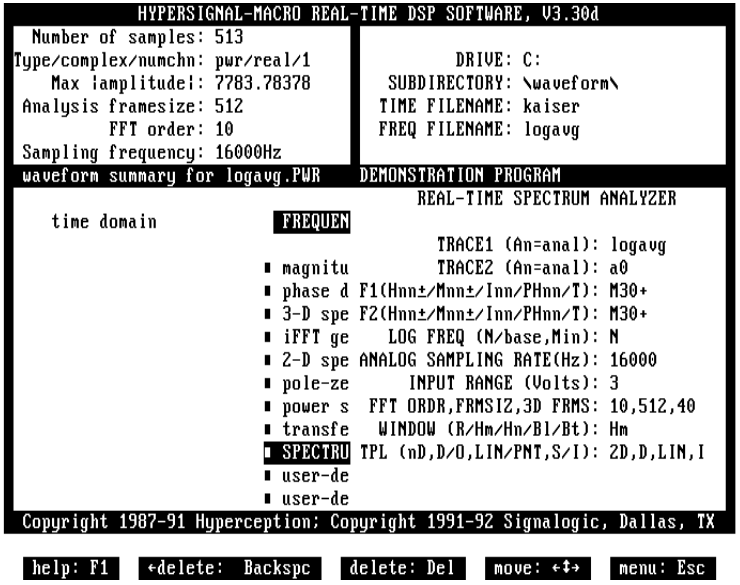


*Figure 11-9*

The menu above directs input from the frequency domain waveform file “logavg”, which was just captured, to be displayed on trace 1 in log magnitude format, and analog input from channel 0 to be displayed on trace 2, also in log magnitude format. Note that the setting of the LOG FREQ field, which is explained below, affects the data displayed in both traces, because both contain a frequency axis. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-10, below:

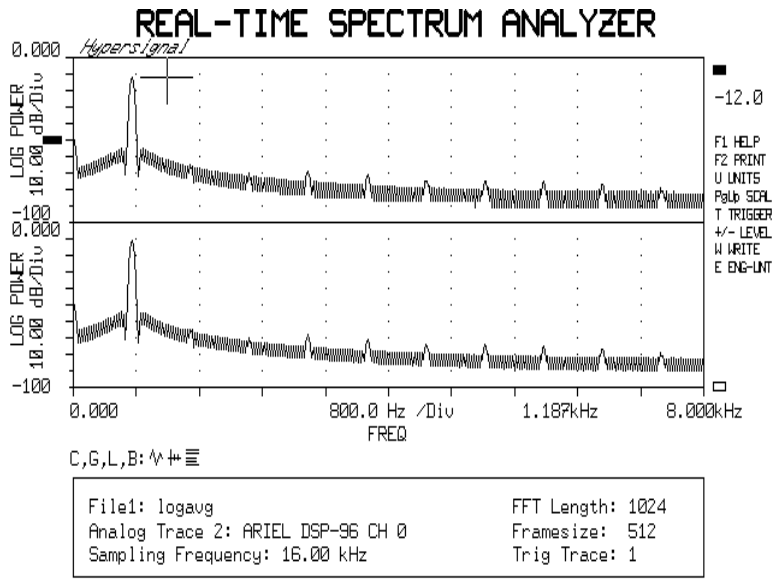


*Figure 11-10*

Trace 1 and trace 2 should be nearly identical, excepting slight variations as trace 2 is continuously updated. In this case we have asked for dual-trace display so the "sameness" of the traces will stand out, but for real work, you might want to highlight the "differences" by using overlaid-trace display (enter 'O' in the FORMAT subfield; see section 11.1.2, Dual- and Overlaid-Trace Display, above). As explained after Figure 11-4 above, pressing the '1' and '2' keys will select the current active trace. All controls work in the same way for waveform file and analog input except for triggering, which is not active in the Spectrum Analyzer for waveform file input (but is in the Digital Oscilloscope; see section 11.6 below for an explanation).

### 11.3.1 Log and Octave Band Frequency Axis Display

For frequency domain data formats (but not for time domain formats), it is also possible to specify a log frequency axis, using the LOG FREQ field. A simple 'N' (none) entry in this field, which is what we have used in all examples up to this point, specifies a linear frequency axis, starting at zero (dc) and ending at half the sampling rate (the maximum bandwidth, de-

pending on the anti-aliasing filters used by the DSP/Analog board). Other entries can be used to specify a base N log axis, or in some cases, special entry such as 1/6 octave band. For example, the partial data-entry menu example below indicates 3 cycles of log 10 display, with a minimum frequency axis value of 200 Hz:

LOG FREQ (N/base,Min): 10^3,200

The miminum frequency axis value is always specified directly in Hz. The next example would produce octave 1/3 band display, this time with a 20 Hz minimum frequency axis value:

LOG FREQ (N/base,Min): OCT^3,20

To toggle easily between linear and log frequency axis modes, simply make the first letter of the LOG FREQ field entry an 'N' or part of a legitimate entry, leaving the rest of the entry in place. The example below, which will invoke linear frequency axis display, demonstrates this:

LOG FREQ (N/base,Min): NCT^3,20

Following is an actual data-entry menu and Spectrum Analyzer display example of averaged frequency format display. For this example, *it is assumed that you have a function generator connected to channel 0 of your DSP/Analog board, and generating a periodic signal.* For the tutorial examples in this section, a sine wave was used. Enter the Spectrum Analyzer menu below:

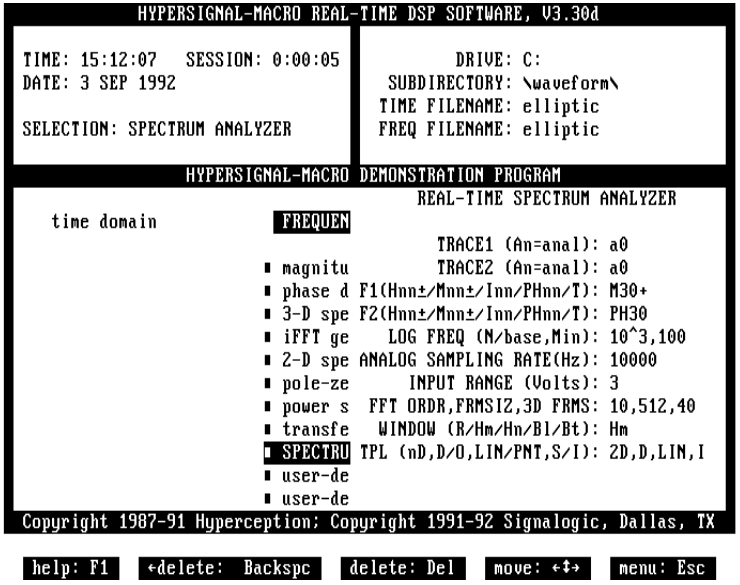


*Figure 11-11*

The menu above directs input from analog channel 0 to be displayed on trace 1 in averaged log magnitude format, and analog input from channel 0 to be displayed on trace 2 in averaged phase format. This entry is identical to the example in Figure 11-7, except that the LOG FREQ field has been set to specify 3 cycles of log 10 frequency data, with a minimum frequency axis value of 100 Hz. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-12, below:

11.3.1 Log and Octave Band Frequency Axis Display

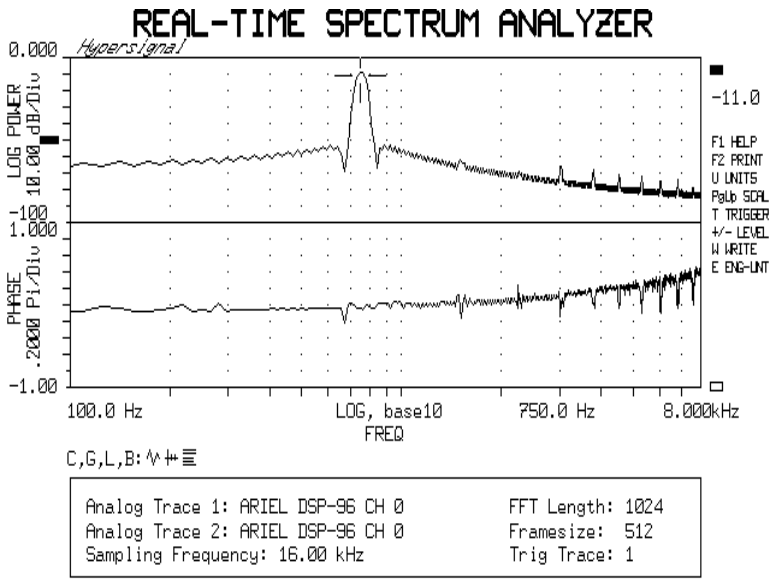


*Figure 11-12*

Note that the grid marks in the display are exactly on multiples of the logarithm base (e.g. ..., 90, 100, 200, ..., 900, 1000, 2000, ..., 9000, 10000, 20000, ..., etc. for base 10) for log frequency axis display. However, the maximum frequency axis value (half the sampling rate) and the minimum frequency axis value (default or specified in the LOG FREQ field) may be any number. After verifying the log frequency axis display in the example above, return to the data-entry menu and enter the LOG FREQ field as follows:

LOG FREQ (N/base,Min): OCT^6,20

The above entry will specify 1/6 octave band display, with a 20 Hz minimum frequency axis value. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer display similar to Figure 11-13, below:

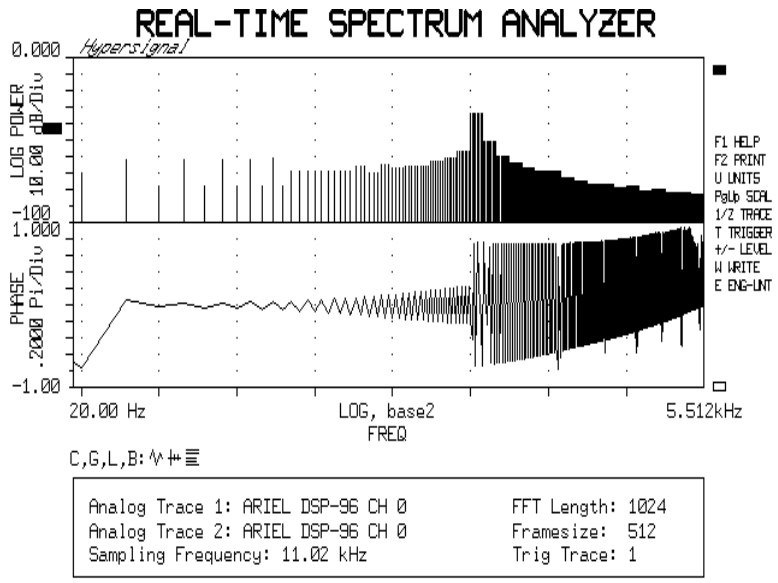


*Figure 11-13*

Again, in the above display the trigger level has been increased somewhat (‘+’ key) to steady the display, especially the phase trace.

1/N octave band display is created by dividing each octave of a log2 frequency axis into N equal sections. Thus, a pure octave band display could be entered in the LOG FREQ field as either:

LOG FREQ (N/base,Min): OCT^1,min

or alternatively,

LOG FREQ (N/base,Min): 2^11,min

where “min” affects only the minimum value of the frequency axis, and might typically range anywhere from 2 Hz to 200 Hz.

**NOTE: Essentially, octave band boundaries are calculated in the Hypersignal Spectrum Analyzer display by repeatedly reducing the *linear* frequency axis by two, starting with half the sampling frequency (bandwidth). Fractional bands are then calculated by segmenting the octave bands evenly. In linear magnitude display modes, the software calculates an arithmetic mean for each fractional band. In log magnitude modes, the software calculates a geometric mean for each fractional band—this calculation is currently being replaced by an arithmetic mean.**

Since a varying number of available FFT output points available for 1/N averaging in each band, the Spectrum Analyzer only draws the resulting averaged value wherever an FFT point is available, so as to depict the frequency resolution of the display clearly. This is why you see "spaces" between points as frequency decreases in Figure 11-13. Note that increasing the FFT length will increase the frequency resolution (and that this can be done independent of the time framesize value). Note also that the crosshair cursor will only land on available FFT points in log frequency axis displays.

## 11.4 Transfer Function Measurement

In the Hypersignal Spectrum Analyzer, transfer function display is calculated by the DSP as the ratio of the complex FFT data from one analog input channel to that of another. Transfer function measurements are often made when attempting to determine the frequency response of an unknown system or circuit: one analog channel is connected to the output of the system (the numerator in the ratio), and the other to the input of the system (denominator in the ratio). Assuming that the mere act of connecting the analog inputs does not change the characteristics of the unknown system, the transfer function ratio can yield the response of the system when the system is supplied with input containing sufficiently wide frequency band content. In the Spectrum Analyzer function data-entry menu, an 'H' format entry in the F1 and/or F2 fields is used to specify transfer function display, which is treated as a frequency domain format. Below is a partial data-entry menu example:

trace 1 displaying the transfer function ratio of channel 0 data over channel 1 data, trace 2 displaying channel 1 data as log magnitude

TRACE1: a0
TRACE2: a1
F1: H20+
F2: M20+

Note the use of the '+' suffix to indicate log magnitude display of transfer function results. No suffix would indicate linear magnitude display, but best results with transfer function display are almost always achieved using log display, because of the large dynamic range resulting from the division intrinsic to transfer function calculation. Also, note the use of averaging—again, best results are usually obtained with some amount of averaging, because the frequency content passing through the system may not be constant.

**POWER NOTE: Transfer function is *always* calculated as the ratio of the input specified by the TRACE1 field over that specified by the TRACE2 field. Thus, if you want to invert the display you can switch these two fields (you could also use 'H-' negative-log entry in the F1 or F2 fields).**

Even though it requires two active analog input channels, transfer function display can also take place with waveform file input, allowing overlay of stored and live input. In the following partial-menu data entry example, the numerator is assumed to be the opposite channel (in a typical 2-channel system) as the TRACE2 input:

trace 1 displaying waveform file input in transfer function format, trace 2 displaying the transfer function ratio of analog channel 0 input over analog channel 1 input

TRACE1: signal
TRACE2: a1
F1: H20+
F2: H20+

**POWER NOTE:** **No transfer function calculation involving waveform file input actually takes place. Rather, the waveform file frequency data, either non-averaged or averaged (as in the above example) is displayed *as if it were the result of transfer function calculation.* This means that the 0 dB reference point is displayed in the middle of the screen, the data is scaled, etc. exactly as are results based on analog data being calculated by the DSP. This means that any waveform file input can be used, although there are three types that would be "technically" correct:**

**1) output from the Filter Construction menu selections**

**2) output from the Transfer Function menu selection**

**3) a waveform file that had been previously stored from a trace in the Spectrum Analyzer displaying analog transfer function data**

Following is an actual data-entry menu and Spectrum Analyzer display example demonstrating transfer function measurement. For this example, *it is assumed that **no** signal is connected to the channel 0 input of your DSP/Analog board*; i.e. the channel 0 input is open. First, enter the following FIR Filter Construction menu:

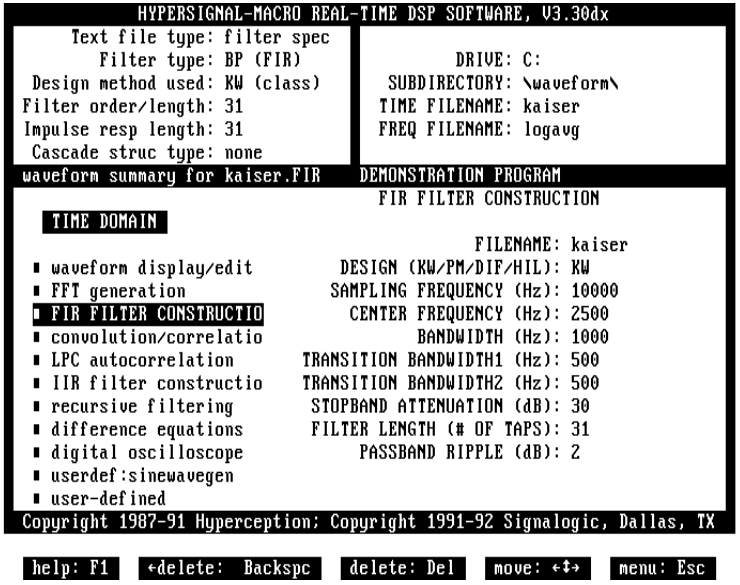


*Figure 11-14*

Press the ‘Enter’ key (or <ctrl-Enter> key sequence) to display the filter graphically. Press the ‘C’ key from the graphical display to construct the filter. A 31-tap Kaiser Window FIR filter will be created, which we will use to demonstrate transfer function display in the Spectrum Analyzer. Now, enter the following Spectrum Analyzer menu:

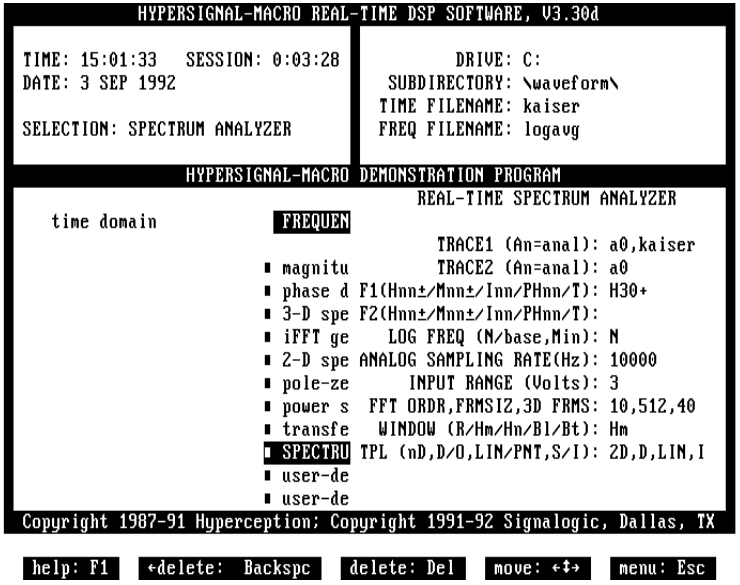


*Figure 11-15*

Note that the “kaiser” filter is active on channel 0 analog input, and that trace 1 will show transfer function display. The output on trace 2 has been disabled by making the first letter of the F2 field entry an ‘N’ (for “none”; blanking the field entirely would also work). Note also that the sampling rate has been set low (10000 Hz), so that the example should run with all DSP boards, including older, slower ones. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer menu similar to Figure 11-16, below:

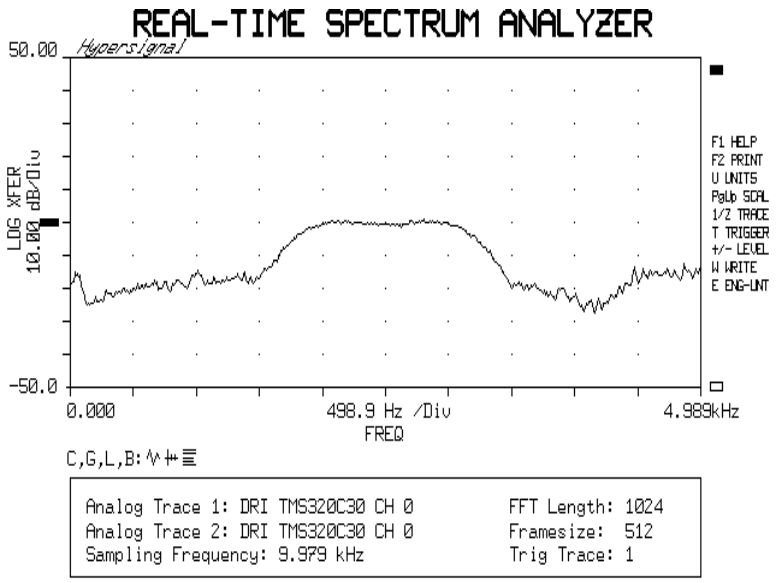


*Figure 11-16*

This display should more or less show the frequency response of the “kaiser” filter. Note that the amplitude level of the passband of the filter should be about 0 dB, which matches the original FIR Filter Construction design. Why does this work? Because without anything connected to the channel 0 input, white noise is more or less the output of the A/D converters. This noise is then filtered by the “kaiser” filter, and due to the ‘H20+’ entry in the F1 data-entry menu field, divided by the unfiltered channel 0 input and averaged for display. Since the A/D converter output basically contains wideband noise, averaging of the transfer function result will produce a facsimile of the “kaiser” filter frequency response.

**POWER NOTE: For transfer function display, the Fourier parameters specified in the data-entry menu apply to the FFT calculation in both the numerator and the denominator. Thus, from a strictly mathematical point of view, it doesn't matter what Fourier parameters are specified, since they will "cancel". However, choice of these parameters is still important based on the nature of the data. For example, often times a Hamming or Blackman window is chosen to prevent wideband frequency domain "edge noise" resulting from time domain energy at or near the edges of the FFT input frames. In the example above this is especially important because the A/D converter output noise is very low-level, and the resulting frequency response could be easily contaminated. Try the above example with no window (a rectangular window) and note the difference!**

Now we will store the transfer function display trace to a waveform file by using the 'W' (write) command. If you have Hypersignal-Acoustic software, ignore the various other types of data output prompts—all we want is the basic power spectra (averaged) magnitude output. Use an entry of 'kaiserX' to answer the filename prompt. Now we must create the power spectra version of the output generated by the original FIR Filter Construction function. Exit the Spectrum Analyzer to the main menu and enter the following FFT Generation menu:

```
                  FILENAME: kaiser
                 FRAMESIZE: 31
                 FFT ORDER: 10
       % FRAMESIZE OVERLAP: 0
     WINDOW (R/Hm/Hn/Bl/Bt): R
  REAL/IMAG OUTPUT (Y/N): N
           PRECISION (I/R): R
  PHASE/MAG APPROX (Y/N): N
```

Remember that the PRECISION and PHASE/MAG APPROX fields only apply if you are using a non-real-time software version, or you have the DSP ACCELERATOR field disabled ('none' entry) in the System Configuration menu. After the above FFT Generation completes, enter the following Power Spectra Generation menu:

```
                  FILENAME: kaiser
       METHOD (A/RA/E): A
  FRAMES TO AVERAGE: 100
```

After the Power Spectra Generation menu completes, enter the following Magnitude Display menu:

```
                    FILENAME1: kaiser
                    FILENAME2: kaiserx
               POWER SPEC (P/M): P
              AMP SQUARE (Y/N): N
          LOG (N/+/-,N/base,Min): +
        Y-LABELS (SCALE,UNITS): 1,Volts
    AMP SCALING (TOP,BOT,REF): 0,0,S
      FORMAT,PLOT (D/O,B/C/D): O,C
                    GRID (N/S/D/I): I
```

In the above menu, we are specifying overlaid-trace display of the “kaiserx” waveform file captured from the Spectrum Analyzer transfer function display, which represents the measured frequency response of the kaiser filter, and the calculated frequency response of the “kaiser” filter. In the display shown in Figure 11-17 below, you can see the similarities and differences.

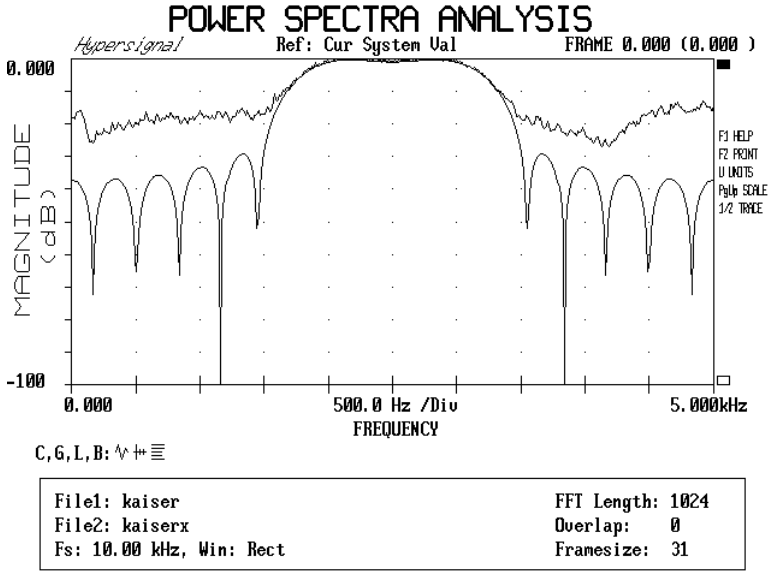


*Figure 11-17*

The “kaiserx” waveform could easily be improved by returning to the Spectrum Analyzer transfer function display and using a sweep generator or pseudo-random noise generator instead of open inputs.

## 11.4.1 Impulse Response Measurement

In the Hypersignal Spectrum Analyzer, impulse response display is calculated by the DSP as the inverse FFT of transfer function calculation. Impulse response measurements can be especially useful when attemping to learn about time domain characterstics, such as delay, of a system. In the Spectrum Analyzer function data-entry menu, an 'I' format entry in the F1 and/or F2 fields is used to specify impulse response display, which is considered to be a time domain format.

**NOTE: The impulse response measurement function may only be installed for some DSP chip families. Check the "HS_REV.DOC" file on distribution diskette #2 to see what DSP chip families are currently working.**

Below is a partial data-entry menu example of impulse response measurement:

trace 1 displaying the impulse response (inverse FFT of the transfer function ratio of channel 0 data over channel 1 data)

TRACE1: a0
TRACE2: a1
F1: I20
F2:

Note the use of averaging—like transfer function magnitude display, best results are usually obtained with some amount of averaging, because the frequency content passing through the system may not be steady.

**POWER NOTE: Impulse response display, by definition, requires transfer function calculation. Thus, all of the notes and rules about transfer function calculation mentioned in section 11.4 above, such as the ratio always being calculated as TRACE1 input over TRACE2 input, apply to impulse response display.**

Following is an actual data-entry menu and Spectrum Analyzer display example demonstrating transfer function measurement. For this example, *it is assumed that **no** signal is connected to the channel 0 input of your DSP/Analog board*; i.e. the channel 0 input is open. It is also assumed that

you have designed the “kaiser” filter used in the examples in the previous section, Transfer Function Measurement. Enter the following Spectrum Analyzer menu:

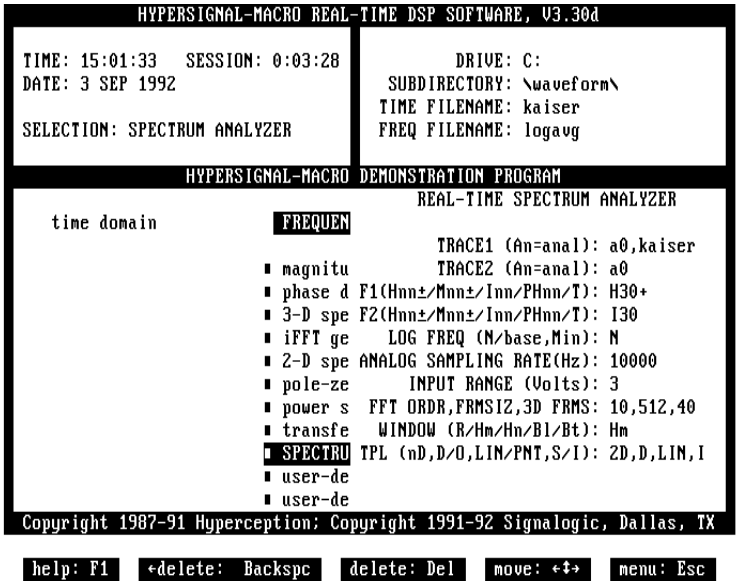


*Figure 11-18*

Note that the “kaiser” filter is active on channel 0 analog input, that trace 1 will show transfer function display, and that trace 2 will show impulse response display. Note also that the sampling rate has been set low (10000 Hz), so that the example should run with all DSP boards, including older, slower ones. After pressing the <ctrl-Enter> key sequence, you should see a Spectrum Analyzer menu similar to Figure 11-19, below:

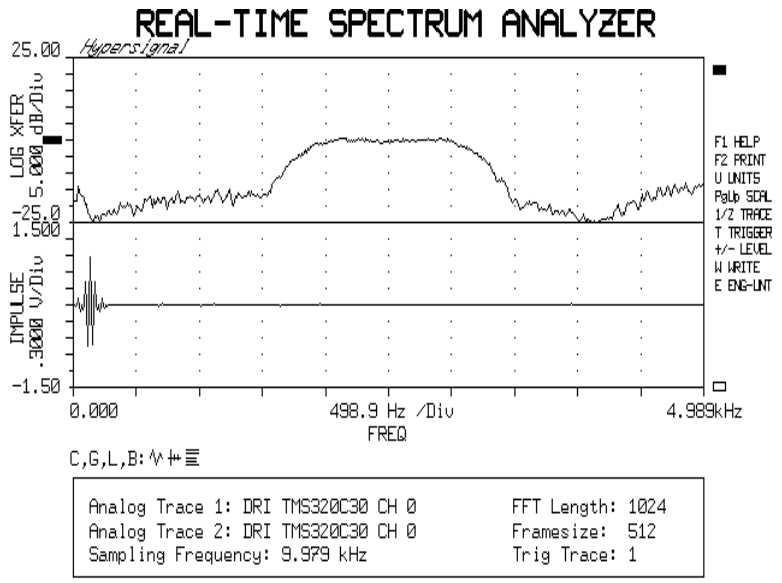


*Figure 11-19*

This display should more or less show the frequency response of the "kaiser" filter on trace 1, and the impulse response of the "kaiser" filter on trace 2. Why does this work? Because without anything connected to the channel 0 input, white noise is more or less the output of the A/D converters. This noise is then filtered by the "kaiser" filter, and due to the 'I20' entry in the F1 data-entry menu field, divided by the unfiltered channel 0 input, inverse FFT'd, and averaged for display. Since the A/D converter output basically contains wideband noise, averaging of the inverse FFT result will produce a facsimile of the "kaiser" filter impulse response.

Now press the '2' key to select trace 2 as the current active trace. Press the 'W' key to invoke the write-to-file command. Answer the prompt with the name 'kaiserX', followed by the 'Enter' key. Because it is a time domain format, captured impulse response output is stored to a time domain waveform file. Exit the Spectrum analyzer and enter the following FFT Generation menu:

FILENAME: kaiserx
FRAMESIZE: 512
FFT ORDER: 10
% FRAMESIZE OVERLAP: 0
WINDOW (R/Hm/Hn/Bl/Bt): R

```
REAL/IMAG OUTPUT (Y/N): N
PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N
```

Remember that the PRECISION and PHASE/MAG APPROX fields only apply if you are using a non-real-time software version, or you have the DSP ACCELERATOR field disabled ('none' entry) in the System Configuration menu. After the above FFT Generation completes, enter the following Magnitude Display menu:

```
FILENAME1: kaiserx
FILENAME2: kaiser
POWER SPEC (P/M): M
AMP SQUARE (Y/N): N
LOG (N/+/-,N/base,Min): +
Y-LABELS (SCALE,UNITS): 1,Volts
AMP SCALING (TOP,BOT,REF): 0,0,S
FORMAT,PLOT (D/O,B/C/D): D,C
GRID (N/S/D/I): I
```

In the above menu, we are specifying dual-trace display of the FFT of the inverse FFT of the Spectrum Analyzer transfer function display (which, although it sounds torturous, simply represents an FFT of the *measured* impulse response of the "kaiser" filter), and the *calculated* impulse response of the "kaiser" filter. In the display shown in Figure 11-20 below, you can see the similarities and differences.

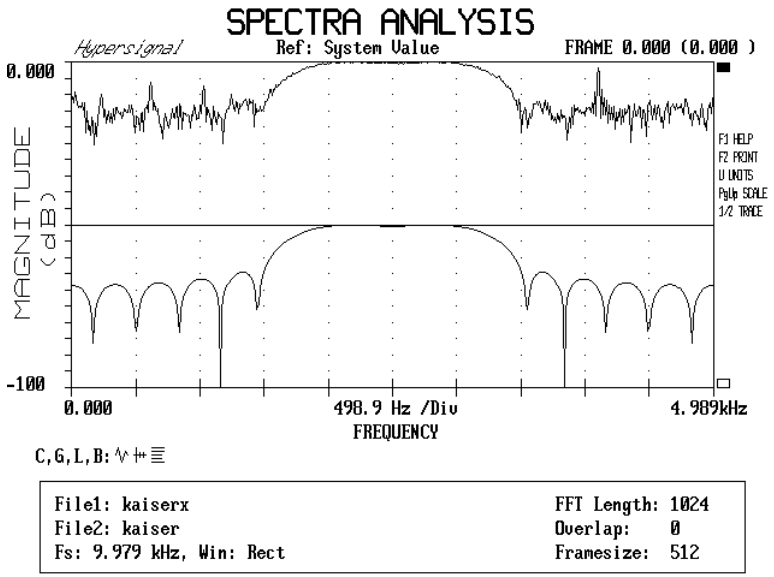


*Figure 11-20*

The “kaiserx” waveform could easily be improved by returning to the Spectrum Analyzer transfer function display and using a sweep generator or pseudo-random noise generator instead of open inputs.

## 11.5 Sampling Rate and Input Range

The sampling rate control in the Digital Oscilloscope and Spectrum Analyzer functions may be the most elemental and important control available. On stand-alone instruments, the sampling rate is usually fixed, or hidden from the user. Or, it may be marginally variable, via controls labeled “wideband”, “narrowband”, etc. In Hypersignal however, because of the tremendous variety of DSP/Analog hardware that is supported, explicit control over the sampling rate must be available. Following is a list of important notes about selection of the sampling rate:

1) The maximum and minimum values of sampling rate are hardware-dependent. Consult the specifications for individual boards and systems in the Hardware Reference Guide. Some hardware can sample as low as a few Hertz; other hardware can only go down to 2 kHz. Some hardware can sample as fast as 10 MHz, some only 20 kHz.

2) Not all hardware allows programmable sampling rates. In these cases, you must enter the ANALOG SAMPLING RATE field in the instrumentation functions to be the fixed rate used by the hardware, even though no information is being communicated to the hardware. Otherwise, the time and frequency units readout in the instrumentation displays will be incorrect. Again, consult the Hardware Reference Guide specifications to see if a fixed rate is in effect.

3) Independent of the rated maximum sampling rate of the A/D and D/A converters used on a given DSP/Analog board or system, *the actual maximum sampling rate possible in the instrumentation functions may be less.* Whether this is true depends on the nature of the DSP chip being used (e.g. its raw speed, interrupt structure and response time, memory access time, etc.), the amount of processing you are currently asking the DSP chip to do (number of channels of input, FFT length, digital filters, etc.), and, in some cases, the speed of your host system I/O bus (for some hardware, a faster bus can decrease the time spent with the DSP chip in hold or in wait-states for external memory access). Consult the "Performance Specifications" section for each entry in the Hardware Reference Guide to see maximum Spectrum Analyzer throughput ratings. The figures in these sections are based on a 33 MHz 80386 system with an 11 MHz I/O bus, *without any digital filters enabled.* The use of digital filters makes determination of the maximum sampling rate problematic, depending on the type and length of the filter. See Users Manual section 4.9.3, Using "Snap-In" Real-Time Filtering, for more information about allowable length of digital filters in the instrumentation displays.

4) DSP/Analog hardware with programmable sampling rates many times cannot always provide the exact rate that you request. In some instances, the hardware will offer between 10 and 20 discrete possibilities, in which case the software will quantize your entry to the nearest possibility. In other instances, the hardware will use a counter-divider technique to attempt to offer a broad range of possibilities; only when the master clock is an integral multiple of the programmed divider will an exact result occur. In all cases, the actual value being used by the software is displayed during operation.

5) In the Spectrum Analyzer function, the sampling rate entry always determines the maximum value of the frequency axis, which will display as half the entered rate (also known as the "Nyquist rate", or "bandwidth"). However, you must be aware of the anti-aliasing filters used on the installed DSP/Analog hardware (if any), which might restrict the bandwidth

to something less than what is shown. You can calculate the minimum bandwidth between the hardware's A/D and D/A converters by doing a stimulus & response measurement; see the examples in chapter 12, STIMULUS & RESPONSE. Again, consult the Hardware Reference Guide for information about anti-aliasing filters and their cutoff frequencies for specific hardware.

The INPUT RANGE field in the Digital Oscilloscope and Spectrum Analyzer functions in most cases should be set to agree with the non-programmable setting of the current DSP/Analog hardware. "Non-programmable" includes gain settings controlled by jumpers, trimpots, and external amplifiers. In such cases, the INPUT RANGE field should be set to a voltage that accurately reflects the ⌠full-scale input range allowed by the hardware. For example, if the hardware is set to convert ⌠3 Volts to ⌠full-scale A/D values (for example, ⌠32767 for 16-bit linear converters), then you would enter a '3' in the INPUT RANGE field. In a few cases, the INPUT RANGE field will actually control programmable-gain circuitry on the DSP/Analog hardware. For example, several boards using the TLC3204x AIC devices can be programmed to one of two or three possible gain ranges using this field. In other cases, continuous programmable-gain entry over a certain range in small increments is possible, and in at least one case the entry is in dB, in 4 dB increments. Consult the Hardware Reference Guide for details in this area for specific DSP/Analog boards and systems.

**NOTE: The ANALOG SAMPLING RATE and INPUT RANGE fields <u>do not</u> apply to waveform file input. If neither TRACEn field specifies analog input, then these two fields will be completely ignored.**

## 11.6 Timebase, Triggering, Vertical Scaling & Position, and Other Controls

In the Digital Oscilloscope, the timebase (amount of time shown, or time per division on the horizontal axis) is controlled by the left and right arrow keys and left and right mouse movement. Left increases the amount of time shown, up to the maximum allowed by the internal Digital Oscilloscope function memory or the DSP board memory. Right, decreases the amount of time shown, down to a few samples. In the Spectrum Analyzer

the timebase is set by the FRAMESIZE field on the data-entry menu; the value cannot be changed from inside the Spectrum Analyzer display. For both instruments, timebase is independent of the current active trace setting in a dual-trace display, and applies to both traces simultaneously.

In dual-trace or overlaid-trace Digital Oscilloscope and Spectrum Analyzer displays, pressing the '1' and '2' keys will toggle the "current active trace" between trace 1 and trace 2. Most controls, including trigger level, vertical position and scaling, crosshair cursor, etc. apply to the current active trace. The current active trace is indicated by two small rectangles at the righthand edge of the display area, with the active trace rectangle appearing solid, or filled-in. Note that a "trigger level indicator" is also displayed for the current active trace (a small rectangle at the lefthand edge of the trace display area).

Trigger level is controlled by the '+' and '-' keys, which apply to the current active trace. The *trigger trace*, though, is controlled by the 'T' key, which toggles the current trigger trace between trace 1 and trace 2 (see note below). A positive trigger value will cause data to be displayed when a level is found that has a rising slope and exceeds the trigger level; a negative trigger level requires a falling slope and a negative data level. Trigger levels within a very small hysteresis area about zero (the default value) direct the program to operate in "free-run" mode, where any data value begins display of the next frame of data.

11.6 Timebase, Triggering, Vertical Scaling & Position, and Other Controls

**NOTE: The following is a very important note about triggering, and might save you some frustrating moments. Even though the trigger level indicator, and the effect of its controls ('+' and '-' keys), change with the current active trace ('1' and '2' keys), the actual *trigger trace* only changes with the 'T' key, which toggles the trigger trace between trace 1 and trace 2. There are various reasons for this, including:**

**1) single-shot operation for a trace is allowed by setting the trigger level, and then switching the trigger to that trace**

**2) the trigger level indicator can be used to mark time domain format amplitude levels**

**3) the trigger level indicator serves as a convenient and noticeable indicator of the current active trace**

**Note that in the Digital Oscilloscope function, the initial trigger trace can be set from the data-entry menu.**

In the Digital Oscilloscope function, triggering is operational for both analog input data and waveform file input data. This is because triggering is essentially a time domain operation, where amplitude levels and slopes are being measured. However, in the Spectrum Analyzer function, triggering does not apply to waveform file input, because the Spectrum Analyzer function does not accept time domain waveform files as input, only frequency domain waveform files.

Vertical scaling is controlled by the 'PgUp' and 'PgDn' keys, which apply to the current active trace. A 'PgUp' key will increase all data formats vertically. Use the the 'PgDn' or 'Enter' keys to return to or restore initial settings. Vertical position is active for time data formats only, and is controlled by the up and down arrow keys and mouse movement. Vertical scaling and position controls cause the amplitude values on the vertical axis to update; the trigger level indicator also moves with such changes, in order to not change the actual trigger level with respect to the input data.

The 'F2' key invokes a pop-up print menu, to allow screen copies to a printer and "EMS memory regeneration" of publication-quality print-outs; see section 5.3, Hardcopy Print-Out.

Other controls include the ‘U’ key, which sets time or frequency axis units; the ‘C’, ‘G’, and ‘B’ keys, which set the waveform data, grid, and background colors, respectively; and the ‘W’ key, which invokes the write-to-file command (see examples in sections 11.3 and 11.4 above).

In frequency domain format displays, arrow keys and mouse movement do not control timebase and vertical position, but instead cause a crosshair cursor, to appear. The crosshair cursor is accompanied by an amplitude readout at the righthand side of the display area, and by a frequency readout beneath the frequency axis. The ‘Home’ key can be used to toggle the cursor crosshairs between solid and partial.

## 11.7 Spectrum Analyzer Calibration

In the Spectrum Analyzer, the crosshair cursor can be used to select a log amplitude value and assign a dB engineering units number to the value in order to calibrate log displays to desired units. Also, other frequency domain displays in Hypersignal contain a log amplitude scaling mode where they, too, can refer to the calibration offset set in the Spectrum Analyzer. When Hypersignal first runs, the default value of the Spectrum Analyzer calibration offset is zero. However, this can easily be changed. Enter the following Spectrum Analyzer menu (it is still assumed that you have a function generator connected to analog input channel 0 of the current DSP/Analog hardware):

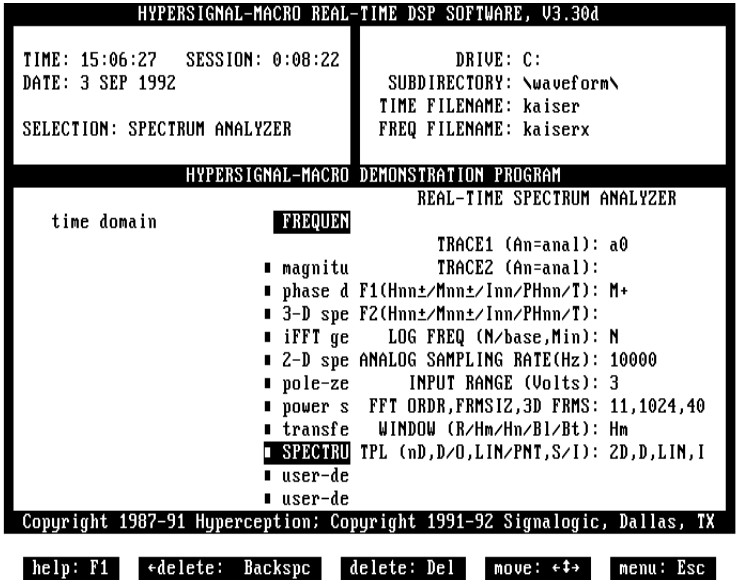


*Figure 11-21*

The above menu will specify channel 0 analog input to be displayed on trace 1 as log magnitude. After pressing the <ctrl-Enter> key, you should see a display similar to Figure 11-22, below:

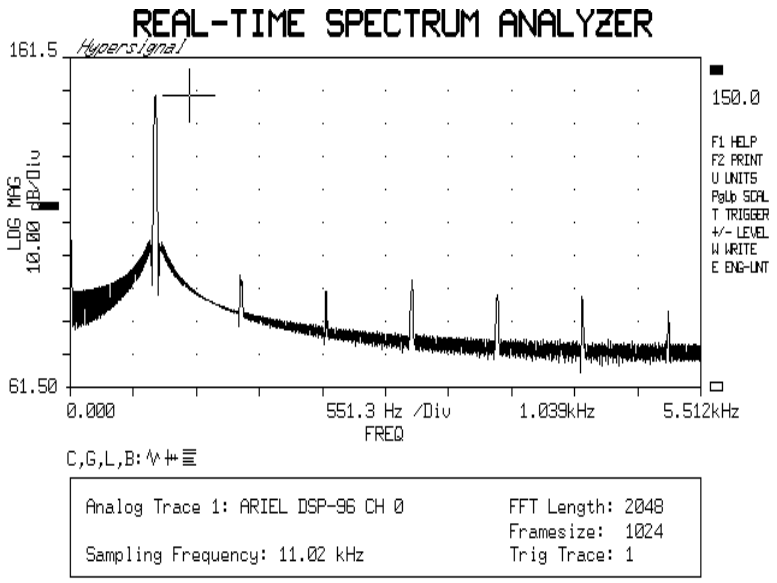


*Figure 11-22*

Use the mouse to position the crosshair cursor so the horizontal crosshair is even with the peak of the tone. Then, press the 'E' key. You should see a prompt appear asking for engineering units. Enter '150' followed by the 'Enter' key. At this point, you have now calibrated the Spectrum Analyzer so the peak of the tone will read out as 150 dB; all vertical axis values and the cursor readout value should update also. Now, press the 'W' key to write this trace to a waveform file. Answer the filename prompt with the entry 'wcal', followed by the 'Enter' key. Now, exit the Spectrum Analyzer to the main menu, and enter the following Magnitude Display menu:

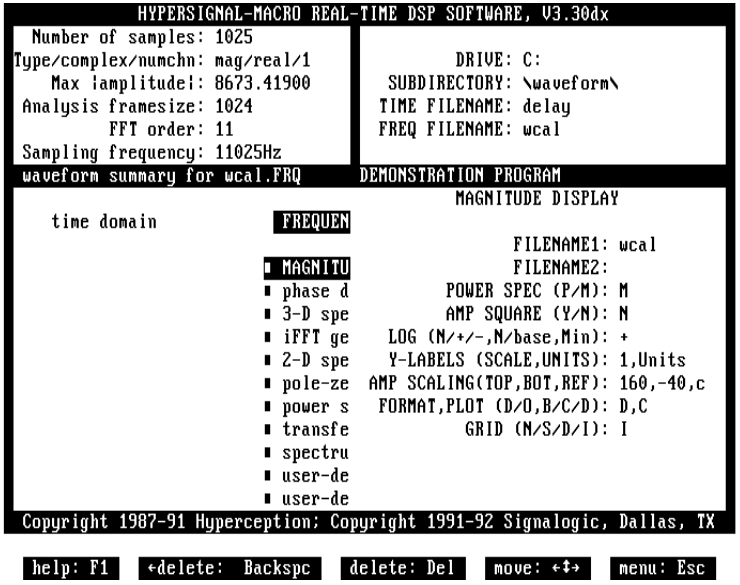


*Figure 11-23*

The above menu specifies the “wcal” waveform file just captured from the Spectrum Analyzer. It also invokes the “system calibration” mode of frequency domain display, in which the calibration offset currently set in the Spectrum Analyzer will be treated as on offset with respect to the system reference value, effectively calibrating frequency domain waveform file-based displays to match the Spectrum Analyzer. Note that the vertical display limits in the AMP SCALING field were selected to encompass the engineering units value that was entered in the Spectrum Analyzer display (150 dB). Also note the ‘C’ entry in the REF subfield of this field, which specifies the system calibration mode of display (for more information on the different selections possible in this field, see Users Manual section 4.5.3.1, Linear vs. Log Frequency Domain Display). After pressing the <ctrl-Enter> key, you should see a display similar to Figure 11-24, below:

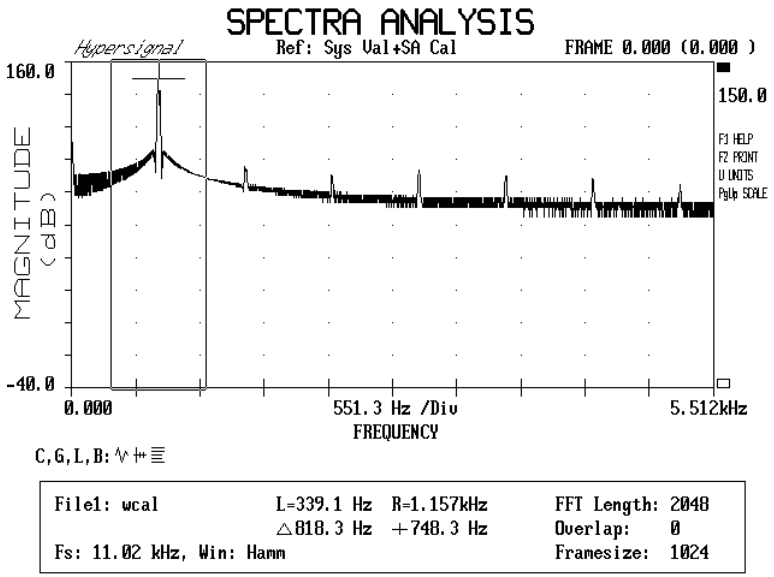


*Figure 11-24*

In the above display, note that the vertical axis ranges from -40 dB to 160 dB, as was specified in the data-entry menu. Also note that the crosshair cursor has been placed exactly on the peak of the waveform, and log amplitude readout at the righthand side of the display area shows 150 dB, which is exactly the engineering units value used to calibrate the Spectrum Analyzer. From this point, if similar menu settings are made in other frequency domain waveform file-based displays (3-D Spectrogram, 2-D Spectrograph), the frequency domain displays in Hypersignal will be calibrated to the real-world phenomenon that was displayed in the Spectrum Analyzer.

**POWER NOTE: To return the Spectrum Analyzer calibration offset to its default value (zero), re-enter the Spectrum Analyzer display, place the crosshair cursor at the very top of the display (as far up as it will go), and press the 'E' key. When the engineering units prompt appears, answer with '0'.**

## 11.8 3-D Map Mode Display

The Spectrum Analyzer also allows a “3-D map mode” of display, which appears similarly to a 3-D Spectrogram waveform file display with the ANGLE field set to 90 degrees, except that the Spectrum Analyzer display is updated continuously, as analog data continues to be input. This type of display is sometimes referred to as a “running waterfall” display, because successive frequency format output frames are drawn slightly behind one another to create the illusion of a third, time dimension. An entry of ‘3D’ in the TPL field will invoke the 3-D map mode. Also, you must have a non-zero number of frames entered in the 3D FRMS field; this value specifies the number of frequency format output frames that will be drawn before the display starts over. Enter the following Spectrum Analyzer data-entry menu (it is still assumed that you have a function generator connected to analog input channel 0 of the current DSP/Analog hardware):

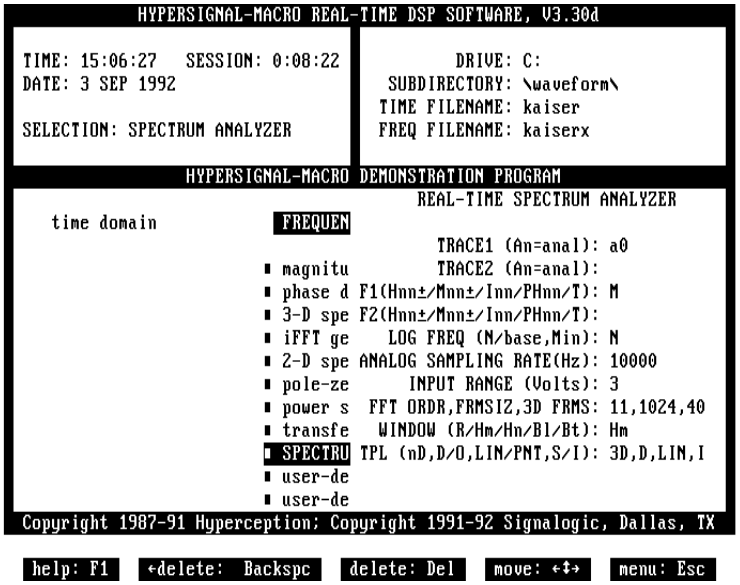


*Figure 11-25*

The above menu specifies input data from analog channel 0 to be displayed on trace 1 in linear magnitude format. Note the ‘3D’ entry in the TPL field (first part of last field). A 3D FRMS value of ‘40’ has been specified, which will cause 40 frequency frames to be displayed on each new 3-D map mode screen. Depending on the type and amplitude of signal you

have connected to the channel 0 input, you should see a display similar to that in Figure 11-26, below:

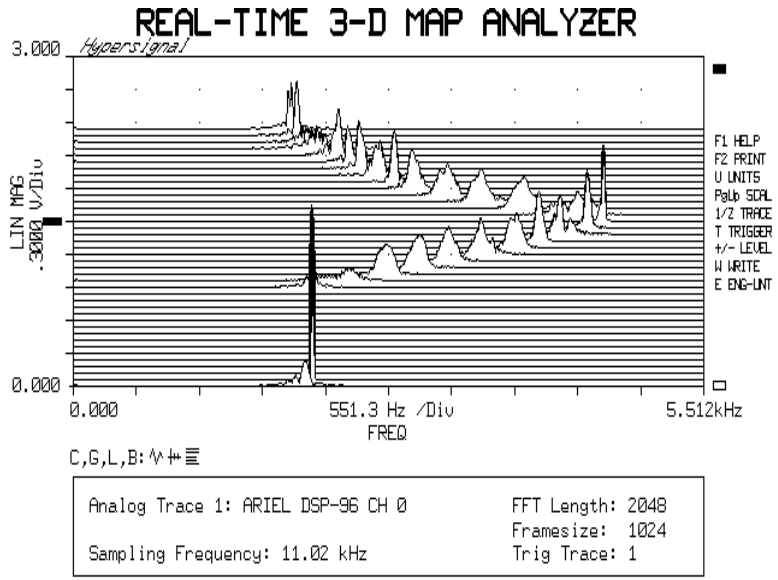


*Figure 11-26*

During the above display, the knob on the frequency generator was vigorously twisted to create an interesting picture. Also, note that linear magnitude was used. Unless the noise floor of a log magnitude display is near the bottom of the trace display area, a 3-D map mode display of log data might be more cluttered than useful. Use the 'PgDn' key to drop the noise floor, and reduce the analog input amplitude slightly for reasonable results in log mode. In the following display, the "signal" waveform is shown in linear magnitude format on trace 1, and analog channel 0 input data in linear magnitude format on trace 2:

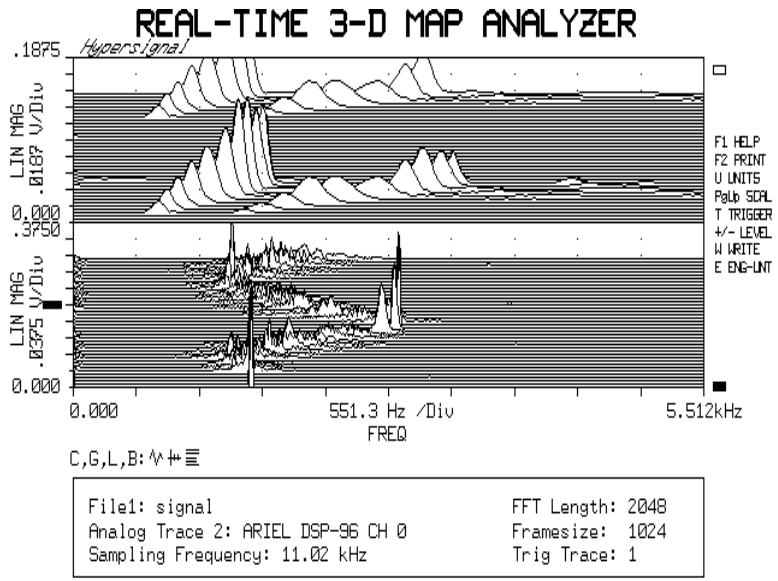


*Figure 11-27*

During the above display, the 'PgUp' and 'PgDn' keys were used to adjust the waveforms to fit well within the available display area.

**POWER NOTE: Use the 'Pause' key on standard PC keyboards to pause the 3-D map mode display as needed.**

The map analyzer will display one or two traces and all combinations of data formats, as in normal operation—even some formats such as time data do not make sense in a "3-D mode". When attempting to obtain print-out from the 3-D map mode, please consult Users Manual section 4.3.2, Printing Waveform Displays.

**NOTE: Remember to set the TPL field back to '2D' in order to return to the standard mode of Spectrum Analyzer operation, in which the software displays data as "two-dimensional" continuously updating traces. When the TPL field is set to '2D', the 3D FRMS field is ignored.**

*this page intentionally left blank*

# 12.0 STIMULUS & RESPONSE

This chapter of the tutorial focuses on using the Analog Conversion menu selection to send out a waveform generated with the Difference Equations function as a “stimulus” to an unknown system, while continuously recording the system’s “response” to an input waveform file. The Analog Conversion function is active only in real-time versions of the software.

In the examples below we first generate “sweep” waveforms using the Difference Equations function, and then use them as output waveforms in stimulus and response operations. After continuously recording the unknown system responses, we can analyze the results with the Convolution/Correlation function to obtain impulse responses, and the FFT Generation and Power Spectra Generation functions to obtain frequency responses.

## 12.1 Generating Stimuli

Consider the basic sine wave generation formula we used in chapter 8, DIFFERENCE EQUATIONS:

$$y[n] = A*\sin(2*pi*n*f/Fs)$$

where f is desired frequency, Fs is sampling frequency, and A is amplitude.

If we desire a sine wave that changes frequency linearly with time, then we must replace the desired frequency “f” in the above equation with a function of n. Therefore, we allow our new argument to the sine function in the above equation to look like:

$$2\pi n(kn)/Fs$$

where k is an unknown constant.

In order to avoid aliasing in the generated waveform, we must ensure that the slope of the changing frequency never exceeds $\pi$ at any time. Thus, by differentiating the above equation with respect to n (the discrete time index), we can formulate the requirement:

$4\pi kn/Fs <= \pi$

To discover k, we must evaluate the above equation at its maximum value of n, which is the last sample in the waveform, or the total length L of the waveform minus 1. This gives:

$k <= Fs/(4(L-1))$

Substituting k back into the equation for y[n] above, we end up with a sweep equation that looks like:

$y[n] = A*sin(pi*n*n/2/(Len-1))$

If we replace Len with "Fr", which will refer to the entered FRAMESIZE field in the Difference Equations data-entry menu (containing the total length of the sweep), subtract some amount from the length to allow for system delay, and add a unit-step multiplier to ensure that zeros are added to the end of the sweep, we can create a sweep stimulus waveform by entering the following Difference Equations menu:

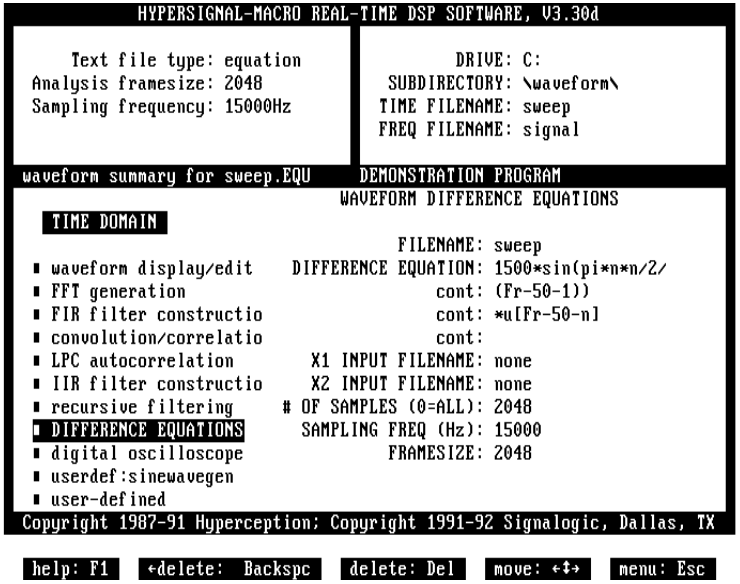


*Figure 12-1*

There are a few interesting notes about the equation in the above menu. First, the equation is independent of sampling frequency. That is, it will sweep all frequencies linearly between 0 and half of the currently pro-

grammed sampling rate for the hardware being used, with a frequency resolution equal to half the sampling rate divided by the “Fr-50" value in the above equation. This, of course, is the second thing to note about the above equation: you can easily control the frequency resolution of the sweep by changing the value of the FRAMESIZE field (referred to in the equation as ”Fr" (although you are also changing the total size of the stimulus and thus the total energy the system under test receives). Third, based on the above analysis, you should not let # OF SAMPLES exceed FRAMESIZE, otherwise aliasing will occur. Finally, in order to allow for system delay in the upcoming post-stimulus analysis, we have purposefully left some zero data (50 samples) at the end of the stimulus through use of the unit-step function (‘u[n]’ in the equation). The unit-step function evaluates to 1 for when its argument is >= zero, and 0 otherwise. The delay has been specified in the equation as a number of samples (50 in the above example) subtracted from FRAMESIZE (‘Fr’ in the equation). At a sampling rate of 15000 Hz, this amounts to a tolerance for about 3 msec of delay; more zeros (or a larger framesize) can be used if the worst-case system delay is expected to be larger. Note that the unit-step function (‘u[n]’ in the equation) has been used to ensure that the the delay period is zero. See the stimulus & response macro example in chapter 15, MACRO EXAMPLES, for a fully-implemented, practical example of this type of sweep generation.

Be sure to allow the above example to execute, because we use the “sweep” waveform again in section 12.2, Simultaneous Input/Output, below. Further applying the above derivation, the following menu would create a waveform that sweeps frequency as the *square* of time, beginning with zero and ending with half the sampling rate:

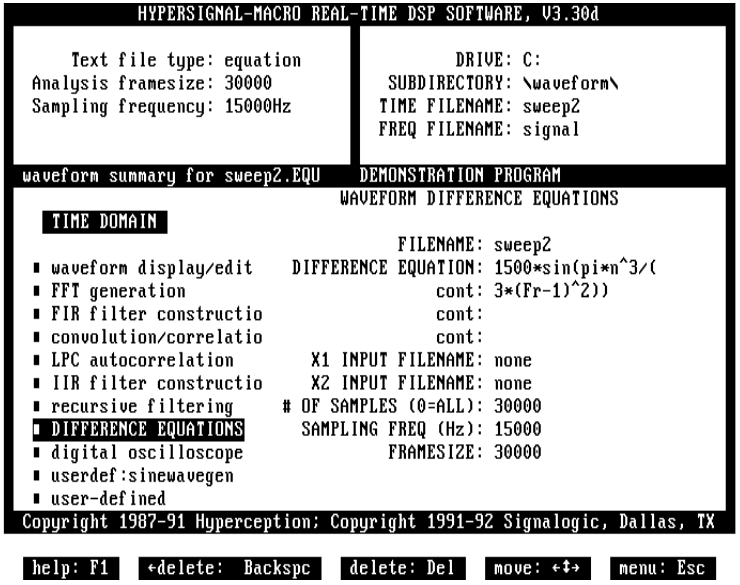


*Figure 12-2*

Generate the above equation, then enter a Waveform Display/Edit menu with “sweep2" as FILENAME1 and FILENAME2 blank. After pressing the ‘PgDn’ key once inside the display, you should see results similar to Figure 12-3, below:

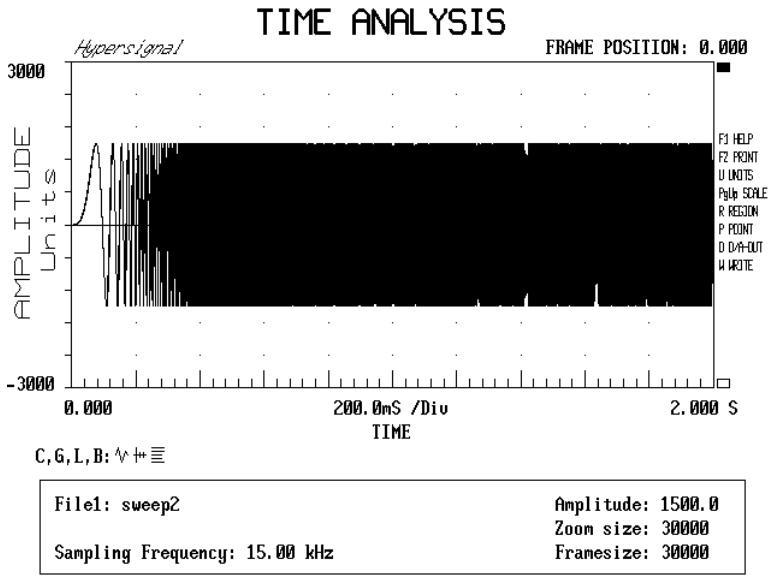


*Figure 12-3*

Note that the waveform does indeed start with a slow frequency and get faster. But looking at 30000 time domain points, or even zooming on a small segment, does not yield a clear picture of what this waveform does with respect to frequency. To get a much better view, we can go into the frequency domain. Enter the following FFT Generation menu:

| | |
|---|---|
| FILENAME: | sweep2 |
| FRAMESIZE: | 1024 |
| FFT ORDER: | 10 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Bl |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to ‘none’ entry). Now enter the following 2-D Spectrograph menu:

| | |
|---|---|
| FILENAME1: | sweep2 |
| FILENAME2: | |
| POWER SPEC (P/M): | M |

```
          AMP SQUARE (Y/N): N
        LOG (N/+/-,N/base,Min): +
AMP SCALING (TOP,BOT,REF): 0,0,A
          COLOR (Cn/Gn/M): C2
            SURFACE (C/R): C
TIME SHOWN (S, 0=ALL): 0
```

Note that we are using a TIME SHOWN entry of ‘0’. A ‘0’ entry invokes the exact length of the sweep2 waveform for display (which should be 2 seconds, since we wrote the equation for 30000 samples at 15000 samples/sec). Upon display, you should be able to clearly see the waveform change frequency (vertical axis) as the square of time (horizontal axis), as shown in figure 12-4, below:

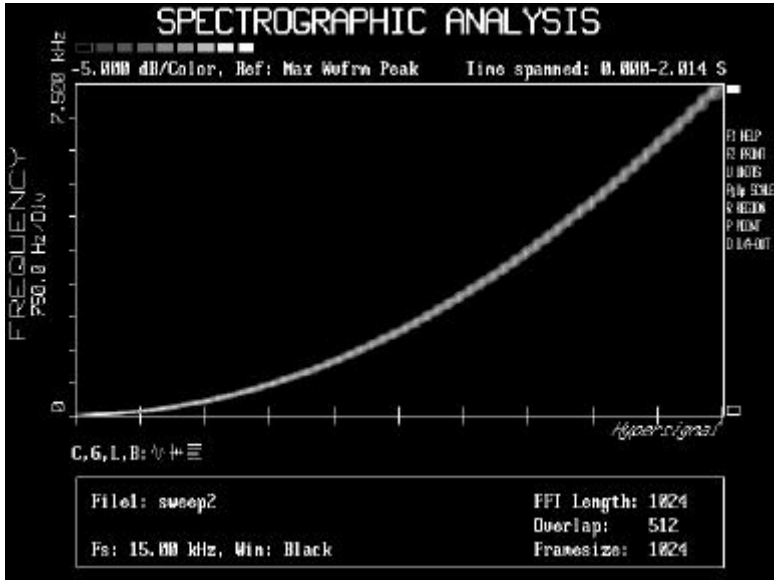


*Figure 12-4*

## 12.2 Simultaneous Input/Output

Now we are ready to actually use our generated waveforms in a stimulus & response operation (presumably to characterize or gather information about an unknown system).

**NOTE: We will assume that you have at least directly connected the channel 0 output to the channel 0 input on your DSP/analog board, if not actually placed some sort of "box" into the circuit. A direct connection, or short, will place the D/A converters and A/D converters of the channel 0 circuitry on the DSP/analog board under test, which is what we have done for this tutorial example.**

When all external connections have been made, and your "system" is set up, enter the following Analog Conversion menu:

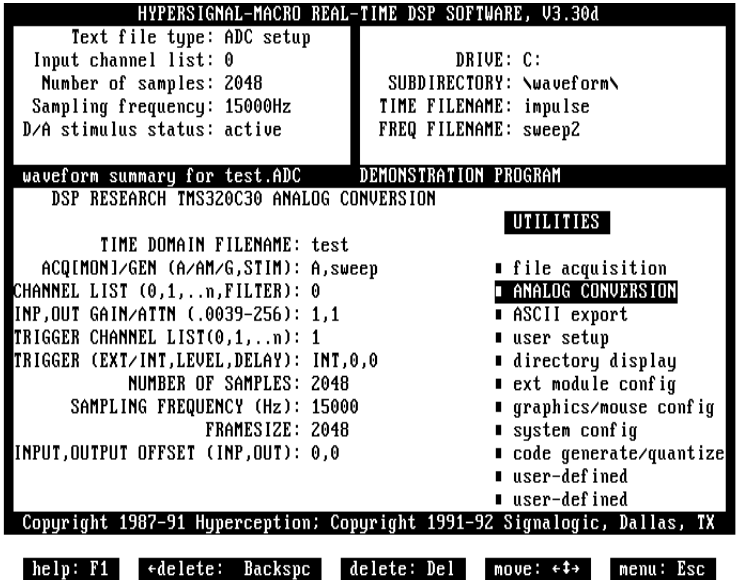


*Figure 12-5*

Please keep in mind that your current DSP board must have enough memory to store the 2048 pt. sweep stimulus, in addition to dual input buffers needed for continuous acquisition. Although we are currently using a small stimulus, in general it is a good idea to equip the DSP board with its manufacturer's expanded memory option if you intend to do much work with stimulus & response. If it should occur that not enough memory is available to hold the entire stimulus waveform, Hypersignal will shorten the input buffers and use whatever memory is left to hold a truncated stimulus.

When this menu is executed, continuous recording will start immediately. The stimulus will be output immediately (because the DELAY subfield has been set to zero), as will recording; both will continue for about 0.1365 seconds (2048 samples / 15000 samples/sec) and then stop. This length of time indicates that one (1) 2048 pt. stimulus frame will be sent through the unknown system and stored to the "impulse" waveform file. Remember that we wrote the "sweep" stimulus equation to leave some room for system delay at the end of each sweep, so the actual stimulus energy duration is about 2000 samples.

**POWER NOTE: A negative trigger value entered for the DELAY subfield would cause the stimulus to begin the entered amount (in seconds) *before* recording begins. A positive DELAY value would cause recording to begin the specified delay period *after* the stimulus begins.**

Note that you can also apply filters, gains, offsets, and trigger criteria to the continuous input, if desired. See chapter 4, ACQUIRING DATA, for more explanation of the fields in the Analog Conversion function. See Users Manual section 4.1.2, Acquiring Waveforms From DSP/Acquisition Hardware, for a fully detailed explanation of this function.

After the above Analog Conversion operation has been completed, enter the following new Analog Conversion menu:

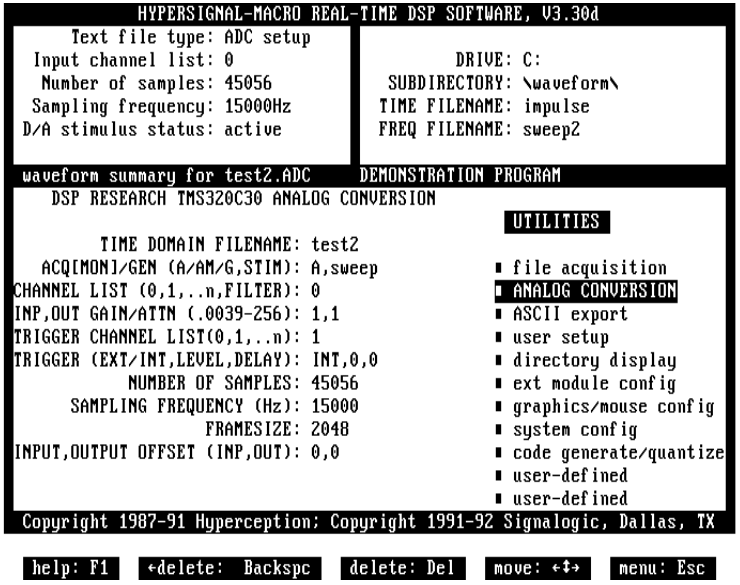


*Figure 12-6*

When this menu is executed, continuous recording will start immediately. The stimulus will be output immediately (because the DELAY subfield has been set to zero), and repeat every 0.133 seconds, while recording continues for about 3 seconds (45056 samples / 15000 samples/sec). This length of time indicates that 22 2048 pt. frames of the stimulus will be sent through the unknown system and stored to the “output” waveform file. Remember that we wrote the “sweep” stimulus equation to leave some room for system delay at the end of each sweep, so the actual stimulus energy duration is about 2000 samples.

## 12.3 Time Domain Analysis

Now that we have a time domain response to our “sweep” waveform, we can cross-correlate the stimulus with the response in order to extract the impulse response and time delay of our unknown system. First, enter the following Convolution/Correlation menu:

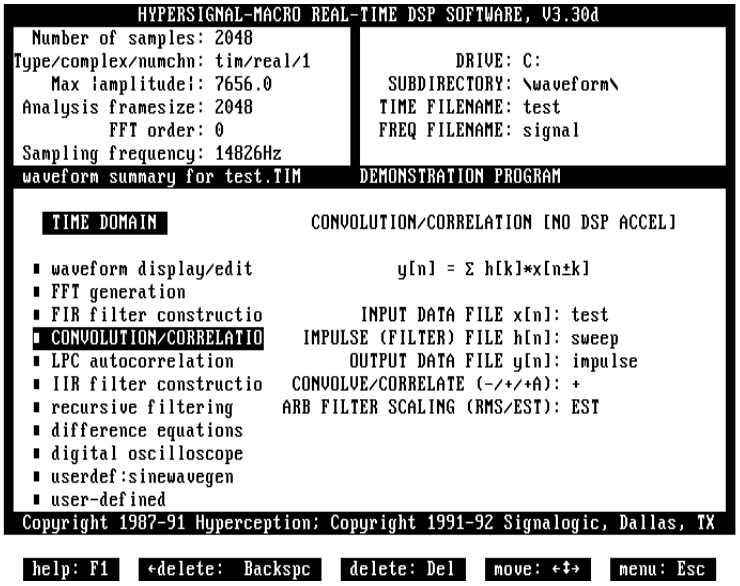


*Figure 12-7*

In the above menu, the '+' entry specifies cross-correlation, and the ARB FILTER SCALING field should be set to 'EST' since the FILTER FILE field contains an arbitrary waveform (as opposed to an output from one of the Filter Construction functions). Note that we are considering the system output, acquired by the Analog Conversion function, as the input waveform, and the sweep stimulus as the filter waveform, which the correlation operation will "slide" past the input waveform. In this case, it is fair to think of the stimulus waveform as a "matched filter", with which we are trying to find the segment of data that best matches it in our input waveform.

**NOTE: The Convolution/Correlation function in Hypersignal is currently limited to a maximum of 16384 samples for the filter waveform (in this case the stimulus waveform). Thus, stimulus waveforms created using the Difference Equations function and intended for cross-correlation time domain analysis should be limited to 16384 points or less.**

After the correlation operation completes, enter the following Waveform Display/Edit menu:

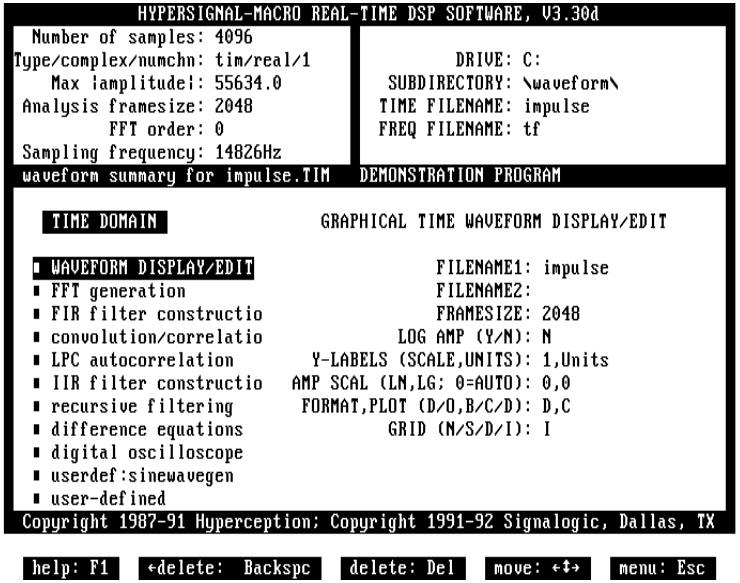


*Figure 12-8*

After entering the last field (or pressing <ctrl-Enter>), you should see a time domain display respresenting the impulse response of the system under test:

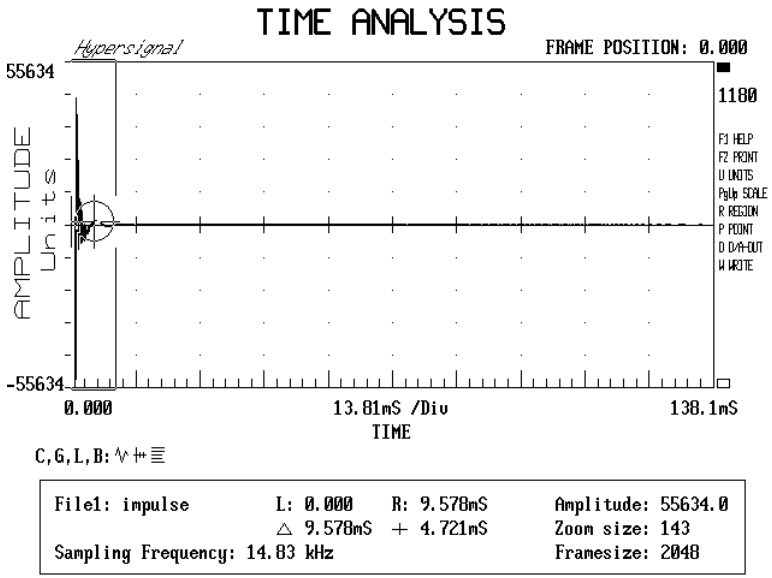


*Figure 12-9*

In this display, any overall delay in the system under test is readily apparent, and can be measured by using the target window (arrow keys or mouse motion) to determine the time between the vertical axis and the onset of the start of non-zero data on the “impulse” trace. Figure 12-10 below shows a zoom-in view based on the position of the target window in Figure 12-9, above:

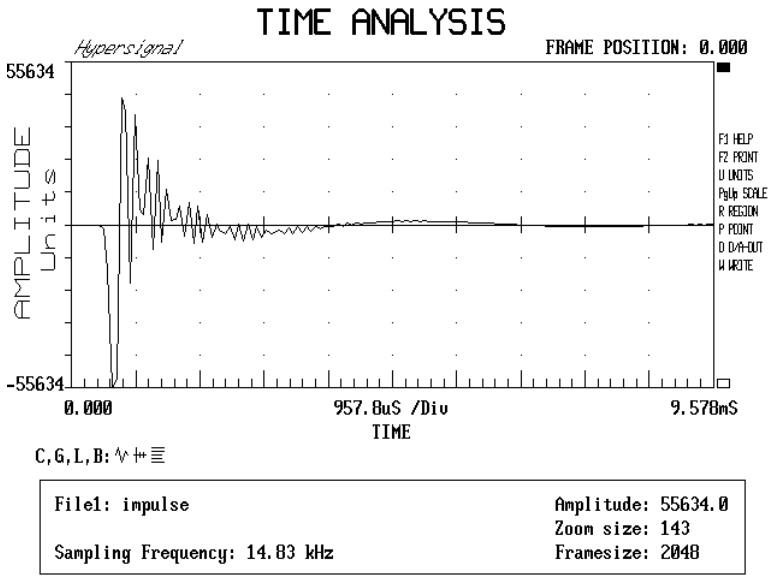


*Figure 12-10*

Remember that in our example, we have connected the output to the input of a particular DSP/analog board, so your display might look completely different. Now enter the following FFT Generation menu:

FILENAME: impulse
FRAMESIZE: 2048
FFT ORDER: 11
% FRAMESIZE OVERLAP: 0
WINDOW (R/Hm/Hn/Bl/Bt): R
REAL/IMAG OUTPUT (Y/N): N
PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to 'none' entry). In the above menu, we have entered 2048 as the FRAMESIZE value, even though the cross-correlation function always slides its "filter" waveform completely past the input, adding the filter waveform length to the input length when creating an output waveform (2048 plus 2048, in this case). Thus, you should see the FFT Generation function create two output frames; the second one is extraneous.

**NOTES: The FFT Generation function in Hypersignal is currently limited to a maximum of 4096 samples per frame. Thus, the impulse response that results from the cross-correlation function should not contain significant energy beyond 4096 samples (should thoroughly decay within this amount of samples). Since a system's response is fixed, regardless of the frequency resolution (length) of the stimulus used, this implies that the original stimulus waveform can be longer than 4096 points, which is indeed the case, but the cross-correlation function currently has a limit of 16384 samples. Thus, stimulus waveforms created using the Difference Equations function and intended for time domain analysis should be limited to 16384 samples or less. An approach is currently under development that avoids length constraints; see section 12.5 below.**

**Note also that we have used a rectangular window and a framesize that encompasses the entire impulse response length (where it has any significant energy), *including any delay that might be present in the unknown system.* Otherwise, the system response would be segmented across two or more FFT frames, which would produce inaccurate results.**

After allowing this operation to complete, enter the following Magnitude Display menu:

```
                FILENAME1: impulse
                FILENAME2:
          POWER SPEC (P/M): M
         AMP SQUARE (Y/N): N
     LOG (N/+/-,N/base,Min): +
  Y-LABELS (SCALE,UNITS): 1,Volts
AMP SCALING (TOP,BOT,REF): 0,0,A
  FORMAT,PLOT (D/O,B/C/D): D,C
            GRID (N/S/D/I): I
```

Entering the last field of this menu (or pressing <ctrl-Enter>), you should see a frequency domain display respresenting the frequency response of the system under test:

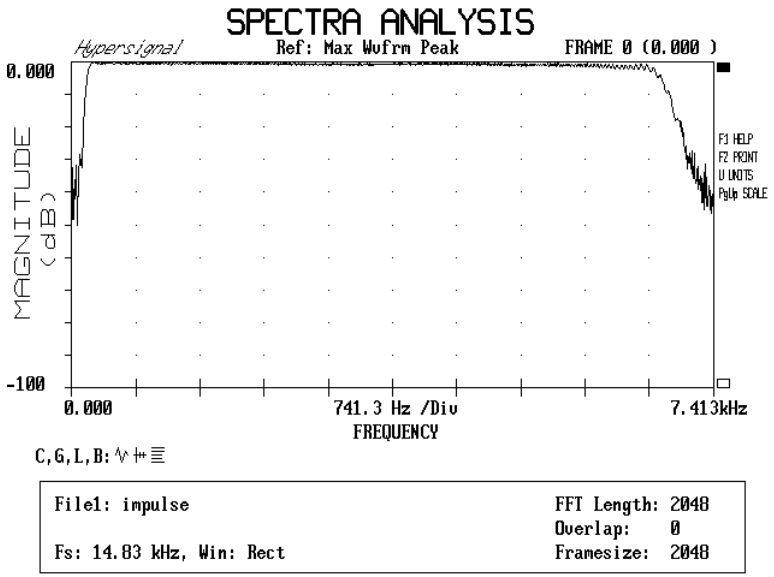


*Figure 12-11*

This display represents the frequency response of the system under test, although we will obtain a more accurate response in the next section using frequency domain averaging. Remember that in our example, we have connected the output to the input of a particular DSP/analog board, so your display might look completely different. For the DSP/analog board used in our example, the A/D converters contained a built-in bandpass function, which is visible in the above display.

**POWER NOTE: The amplitude of the "impulse" frequency domain waveform is not correct—because the original time domain waveform has not been labeled as a transfer function waveform. The cross-correlation function simply transferred the attributes of the input waveform to those of the output, which indicated general-purpose time data. If you use the external program "headitor" to edit the waveform header of the "impulse.tim" waveform, and set its type attribute to "H[n]" (transfer function), a subsequent FFT Generation function should produce a more correct result, amplitude-wise. However, since the original filter waveform in the cross-correlation function was not a product of the built-in filter construction functions, the cross-correlation function could only estimate its gain. For a detailed explanation of the different ways the program interprets amplitude information for various types of data in the frequency domain, see Users Manual section 4.5.3.2, Frequency Domain Display Scaling. For a detailed explanation of the estimation of amplitude for arbitrary time data filters, see Users Manual section 4.9.2, Applying Filters with Convolution.**

## 12.4 Frequency Domain Analysis

By transforming our longer output waveform into the frequency domain, we can perform analysis to get a better look at the unknown system's frequency response. First, enter the following FFT Generation menu:

```
                    FILENAME: test2
                   FRAMESIZE: 2048
                   FFT ORDER: 11
     % FRAMESIZE OVERLAP: 0
     WINDOW (R/Hm/Hn/Bl/Bt): R
REAL/IMAG OUTPUT (Y/N): N
          PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N
```

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to 'none' entry).

**NOTE: It is highly suggested that you use a rectangular window, no overlap, and a framesize value that encompasses the stimulus length when producing the FFT Generation waveform for the system output. Segmenting the system output waveform can cause phase discontinuity problems and misleading magnitude display results. See chapter 13, TRANSFER FUNCTION ANALYSIS, for information concerning situations where segmentation is valid.**

Now enter the following Power Spectra Generation menu:

```
                FILENAME: test2
          METHOD (A/RA/E): A
    FRAMES TO AVERAGE: 100
```

The FRAMES TO AVERAGE field is interpreted as a percentage when straight averaging is used ('A' entry in METHOD field). Thus, the '100' entry in the FRAMES TO AVERAGE field specifies all 22 input frames to be averaged together. After allowing this operation to complete, enter the following Magnitude Display menu:

```
               FILENAME1: test2
               FILENAME2:
          POWER SPEC (P/M): P
         AMP SQUARE (Y/N): N
     LOG (N/+/-,N/base,Min): +
  Y-LABELS (SCALE,UNITS): 1,Volts
AMP SCALING (TOP,BOT,REF): 0,0,A
  FORMAT,PLOT (D/O,B/C/D): D,C
          GRID (N/S/D/I): I
```

Entering the last field of this menu (or pressing <ctrl-Enter>), you should see a frequency domain display respresenting the averaged frequency response of the system under test:

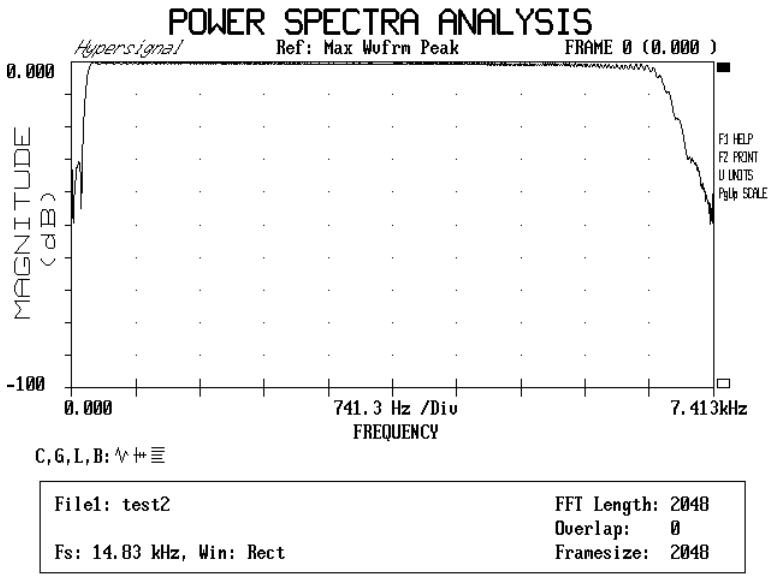


*Figure 12-12*

This display represents an averaged frequency response of the system under test, resulting from repeated passes of the stimulus through the system under test. Remember that in our example, we have connected the output to the input of a particular DSP/analog board, so your display might look completely different. For the DSP/analog board used in our example, the A/D converters contained a built-in bandpass function, which is visible in the above display.

## 12.5 Extended Time Domain Analysis

A new time domain analysis approach to analyze stimulus and response results is currently being implemented in Hypersignal-Acoustic software, in conjunction with two acoustic experts who have been kind enough to share their ideas with Signalogic. The new approach does not suffer from length limitations imposed by the Convolution/Correlation function, and can be implemented as a single-step macro. For more information on this approach as it develops, please contact Signalogic.

*this page intentionally left blank*

# 13.0 TRANSFER FUNCTION ANALYSIS

So far in the tutorial, we have used known stimuli in stimulus & response operations to characterize unknown systems. But what if *both* the input and output are unknown? Transfer function analysis allows one to "tap off" from any two points in a system, and determine the transfer function of the data transmission path between the two points (if all frequencies are present, or eventually present, in the input to the transmission path).

Hypersignal-Macro contains two different transfer function analysis functions:

1) file-based "Transfer Function" frequency domain menu selection

2) Spectrum Analyzer real-time transfer function display (real-time software only)

The Transfer Function menu selection takes two sets of previously created real and imaginary frequency domain waveform files (".IMA" and ".REA" filename extensions, and generates a new real and imaginary waveform set by performing a frequency domain complex division of the system output by the system input. Also, magnitude and phase waveform files are generated as well, for subsequent use in standard frequency domain displays.

**NOTE: You must have performed a previous FFT Generation operation on each waveform with the REAL/IMAG OUTPUT menu field enabled. The FFT analysis parameters used for each waveform should be identical.**

See Users Manual section 4.7.4.1, Transfer Function Analysis, for fully detailed information about the file-based method. The Spectrum Analyzer instrumentation display function contains a real-time transfer function display for most supported DSP hardware. See chapter 11, SPECTRUM ANALYZER, for examples data-entry menus to set up transfer function display, and its relative, impulse response display.

Now, we will take the “sweep” and “test” waveforms that we generated and worked with chapter 12 to illustrate the Transfer Function menu function. Note that the outcome is pre-ordained in this case, since we know that the “sweep” input, when used in the denominator of the complex division, will represent a case of simply “dividing by 1", since the sweep waveform contains all frequencies, by definition. However, this will still serve as a good how-to example.

**NOTE: It is important to remember that Transfer Function Analysis, as compared to stimulus & response, represents a general case: the “numerator” and “denominator” waveforms can come from anywhere. However, it is equally important to remember that you won’t get a complete picture of the unknown system or transmission path unless all frequencies are present in the input to the system. With unknown data, this can mean recording a lot of output data, and performing averaging.**

First, we must regenerate the FFT output for the “sweep” and “test” waveforms with the REAL/IMAG OUTPUT field in the FFT Generation menu enabled. This will allow us to run the Transfer Function menu selection, which will generate magnitude and phase waveform files. The Transfer Function selection will calculate the ratio of the “test” waveform over the “sweep” waveform (output divided by input). Enter the following FFT Generation menus, allowing each one to execute:

```
                    FILENAME: sweep
                  FRAMESIZE: 2048
                   FFT ORDER: 11
  % FRAMESIZE OVERLAP: 0
  WINDOW (R/Hm/Hn/Bl/Bt): R
REAL/IMAG OUTPUT (Y/N): N
          PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N
```

```
                    FILENAME: test2
                  FRAMESIZE: 2048
                   FFT ORDER: 11
  % FRAMESIZE OVERLAP: 0
  WINDOW (R/Hm/Hn/Bl/Bt): R
REAL/IMAG OUTPUT (Y/N): N
          PRECISION (I/R): R
```

PHASE/MAG APPROX (Y/N): N

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to 'none' entry).

**POWER NOTE: It is highly suggested that you use a rectangular window, no overlap, and a framesize value that encompasses the stimulus length when producing the FFT Generation output waveforms that will be used as input to the Transfer Function menu selection. Segmenting the input waveforms can cause phase discontinuity problems and misleading magnitude results in the Transfer Function output. Segmentation can be valid if each FFT frame contains "independent" data; for example, a swept sine input that repeated every 4096 samples could be processed in the FFT Generation function with a framesize of 4096. Transfer Function analysis on the resulting complex frequency waveforms would then generate a series of system response frames that could be viewed in the 3-D Spectrogram or 2-D Spectrograph display functions.**

Next, enter the following Transfer Function menu:

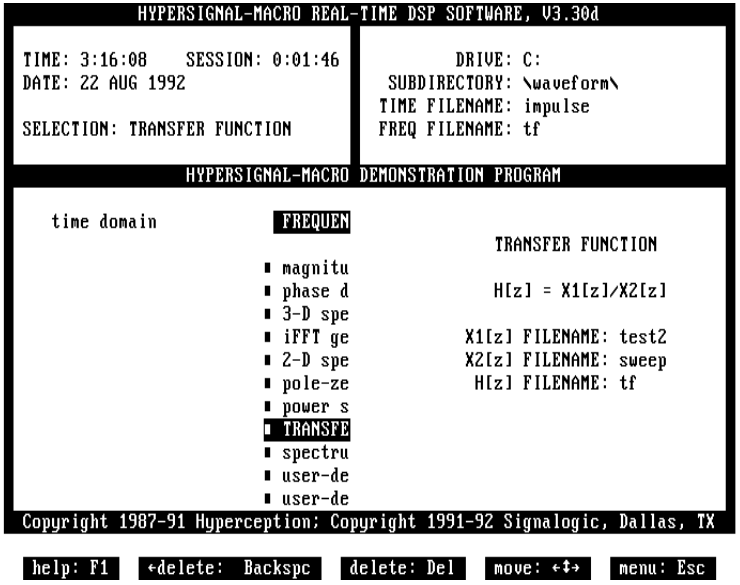


*Figure 13-1*

If the real & imaginary waveform files for either of the “sweep” or “test” waveforms do not exist, then an error message will be displayed and you will have to re-execute an FFT Generation function(s) with the REAL/IMAG OUTPUT field enabled (set to ‘Y’).

The Transfer Function selection will generate new real and imaginary data waveforms, as well as magnitude and phase waveforms, containing the transfer function result. Thus, using the new, “tf”, transfer function filename, all Hypersignal frequency domain displays can immediately be employed to view the transfer function results. Enter the following 3-D Spectrogram menu to see the results of the transfer function operation:

```
                FILENAME1: tf
                FILENAME2:
          POWER SPEC (P/M): M
         AMP SQUARE (Y/N): N
      LOG (N/+/-,N/base,Min): +
    SURFC, GRID(H/M/N,N/S/I): H,N
    Y-LABELS (SCALE,UNITS): 1,Units
AMP SCALING (TOP,BOT,REF): 0,0,A
ANGLE,TIME SHOWN (5-90, S): 45,0
```

After entering the last field, or pressing <ctrl-Enter>, you should see a display similar to Figure 13-2, below:

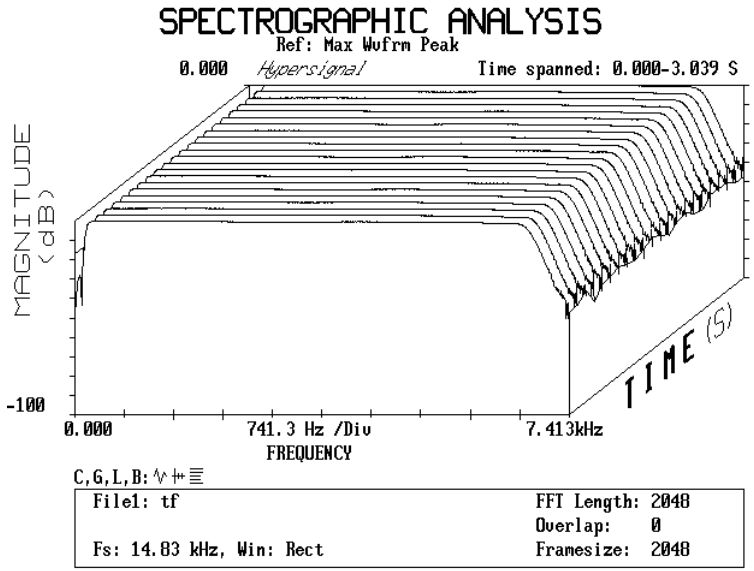


*Figure 13-2*

In this example, the input to the Transfer Function menu selectin was the “test2" waveform, which contains several FFT output frames, and can be averaged to generate a more accurate response. Enter the following Power Spectra Generation menu:

FILENAME: tf
METHOD (A/RA/E): A
FRAMES TO AVERAGE: 100

The FRAMES TO AVERAGE field is interpreted as a percentage for straight averaging (‘A’ entry in the METHOD field), which means an entry of ‘100’ will average all frames. After this function completes, enter the following Magnitude Display menu:

FILENAME1: tf
FILENAME2:
POWER SPEC (P/M): P
AMP SQUARE (Y/N): N
LOG (N/+/-,N/base,Min): +
Y-LABELS (SCALE,UNITS): 1,Volts
AMP SCALING (TOP,BOT,REF): 15,-85,S

FORMAT,PLOT (D/O,B/C/D): D,C
GRID (N/S/D/I): I

Note that we have allowed for some “headroom” in the display, entering 15 dB as the maximum vertical-axis value. This is because of the particular system that has been analyzed in the tutorial example, which exhibited significant gain. Also note the ‘S’ entry in the REF subfield, which specifies that unity gain should appear at 0 dB (see Users Manual section 4.5.3.1 for a better explanation). After entering the last field of this menu (or pressing <ctrl-Enter>), you should see a frequency domain display respresenting the averaged frequency response of the system under test, as shown in Figure 13-3, below:

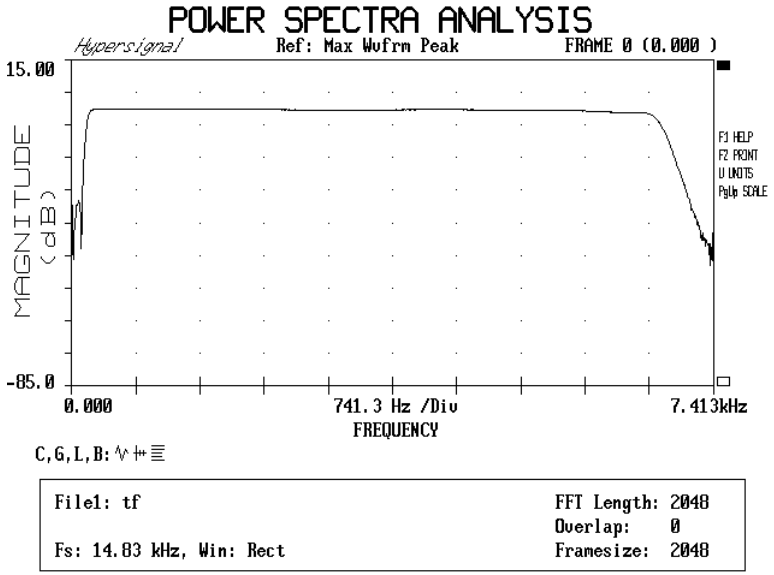


*Figure 13-3*

This display is similar to the “impulse” frequency response that was calculated in section 12.4, Frequency Domain Analysis, as one of the examples for stimulus & response. There is essentially no difference because the original stimulus “sweep” waveform has a flat frequency response, resulting in division by 1 in the Transfer Function menu selection.

**POWER NOTE:** **You may have noticed that the amplitudes of the "impulse" frequency domain waveform calculated in section 12.4 and the "tf" frequency domain waveform calculated here are different, which imply conflicting gain results for the transfer function of our system under test. This is because the "impulse" waveform was originally the output of a cross-correlation function based on general time data inputs, and was not labeled as a transfer function waveform, as was the "tf" waveform. If you use the external program "headitor" to edit the waveform header of the "impulse.frq" waveform, and set its type attribute to "H[n]" (transfer function), the amplitudes should then agree. For a detailed explanation of the different ways the program interprets amplitude information for various types of data in the frequency domain, see Users Manual section 4.5.3.2, Frequency Domain Display Scaling.**

*this page intentionally left blank*

# 14.0 HILBERT TRANSFORMS

This chapter considers the design of Hilbert Transform filters as FIR all-pass delay lines, and then uses the Waveform Display/Edit function to measure the phase-shift effects of the Hilbert Transform, and the Digital Oscilloscope function to do the same, except in real-time. This chapter combines several Hypersignal functions, including filter design, convolution, and instrumentation display, and highlights the flexibility and ease-of-use of the program.

## 14.1 Hilbert Transform Design Example

First, we must design the Hilbert Transform filter. Enter the following FIR Filter Construction menu:

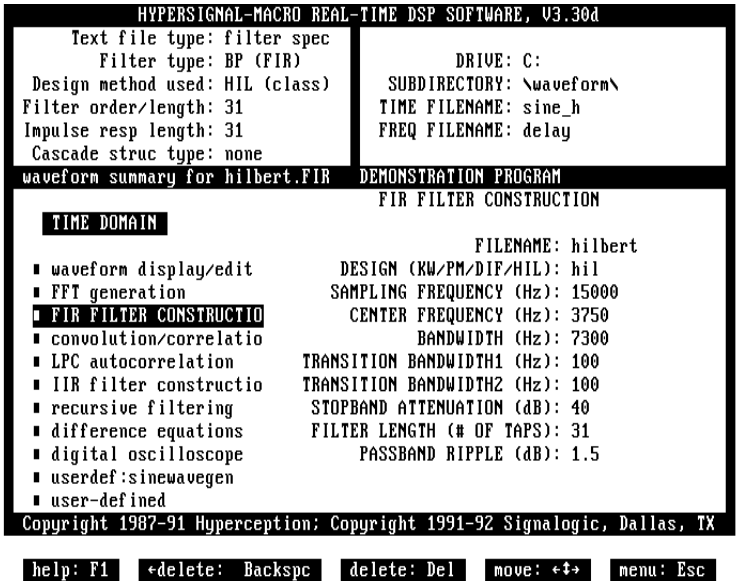


*Figure 14-1*

This is a basic Hilbert Transform design. If you are not already familiar

with the individual fields of the Filter Construction menus, then please refer to chapter 9, DESIGNING FILTERS, above. For detailed information on the "why" of the above field entries, see Users Manual section 4.8.1.7, Designing Hilbert Transform Filters. The 'HIL' entry in the DESIGN field invokes the Hilbert Transform subset of the Parks-McClellan design method. While the cursor is positioned on the last field, press the 'Enter' key (or any button on the mouse) to advance to the "graphical filter design" menu. You should see a basic outline of the desired filter shape relative to half the sampling rate (Fs/2, or 7500Hz). Note that transition bandwidth areas are "undefined" (no outline)—this is done to allow the Parks-McClellan design algorithm to achieve optimum length by concentrating error minimization efforts only on passband and stopband areas.

**POWER NOTE: If, for some reason, you needed to edit or change a transition area, you could place the target box around it (or any undefined region) and "define" it at any time. See Users Manual section 4.8.2, Designing Arbitrary Filters In Hypersignal, for more information on editing arbitrary filters.**

If the passband area is centered on the graphical display area, then the filter specifications have probably been entered correctly. Now press the 'C' key to construct the filter. If you have been through this example before, you may see a prompt asking permission to overwrite the existing "hilbert" files; answer 'Y' to this prompt.

Please note here that the program estimates the required length of the FIR filter to meet your specifications, and also odd or even-adjusts the length, if necessary, depending on the type of filter (highpass, bandstop, etc.). If the estimated length is different than the length you've entered (plus or minus one) then the program will prompt you for which length to use. This example should take 4 "Remez-Exchange" iterations to complete, which could be anywhere from a few seconds to a few minutes, depending on your machine speed and the existence of a math coprocessor.

Next, we'll design an all-pass filter with a delay exactly the same as the Hilbert Transform filter. Enter the following FIR Filter Construction menu:

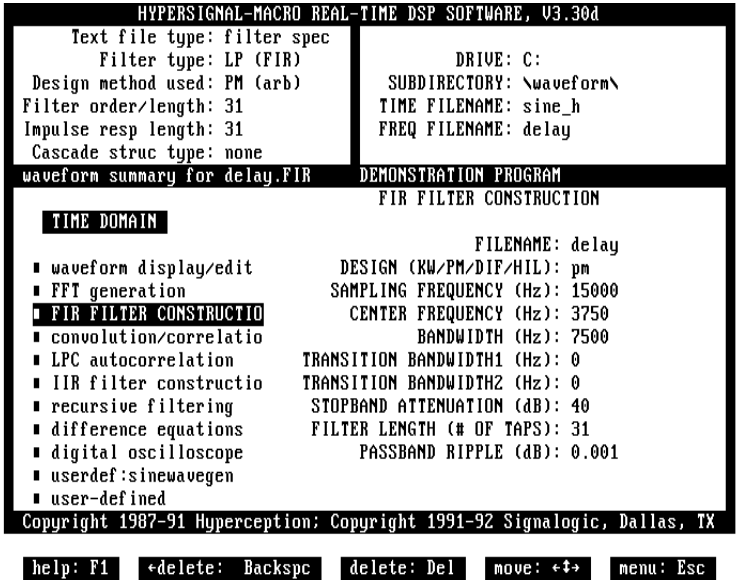


*Figure 14-2*

This is a basic Parks-McClellan design. Note that the passband area stretches from dc (zero) up through half the sampling rate, which means that the filter is all passband, with no transition bands (the transition bands could have been entered to any value in this example because the width of the passband causes them to be ignored). As we did above for the “hilbert” design above, go to the graphical design menu, and press ‘C’. The program will ask if you would like to use a certain (probably large) number of taps; for this example answer the prompt with a ‘N’. This filter should require about 100 iterations to complete.

**POWER NOTE: Note that delaying a waveform in Hypersignal can actually be accomplished in one step by using the Difference Equations function. A difference equation of**

**y[n]=x1[n-M]**

**can can be used to delay the x1 waveform by M samples. The reason why we use a Filter Construction function in this example is to produce output suitable for downloading to a DSP board and running in real-time.**

Now we will use the Convolution/Correlation and Waveform Display/Edit functions, and then the real-time Digital Oscilloscope function to see the Hilbert Transform in action. If you are using a real-time version of software, then you might want to skip the following Waveform Display/Edit Verification section and go right to the Digital Oscilloscope section, since it is an easier and faster method for verifying our example Hilbert Transform filter, and for verifying digital filters in general.

## 14.2 Waveform Display/Edit Verfication

First, we must apply the filters. Choose the Convolution/Correlation function and enter the following menu:

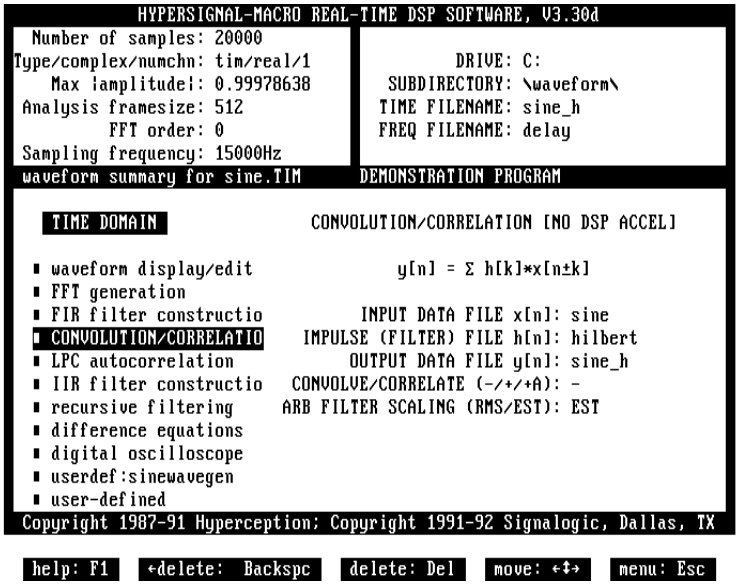


*Figure 14-3*

Note that we assume here that you have previously created the “sine” waveform using Difference Equations (see chapter 8, DIFFERENCE EQUATIONS). While the cursor is positioned on the last field, pressing the ‘Enter’ key (or any button on the mouse), the output filtered waveform “sine_h” will be created.

Second, apply the delay line filter using the following Convolution/Correlation menu:

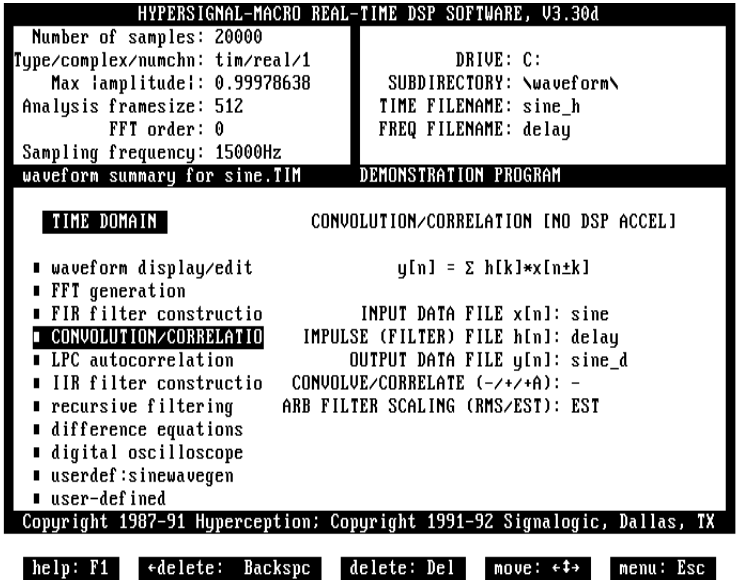


*Figure 14-4*

Now enter the following Waveform Display/Edit menu:

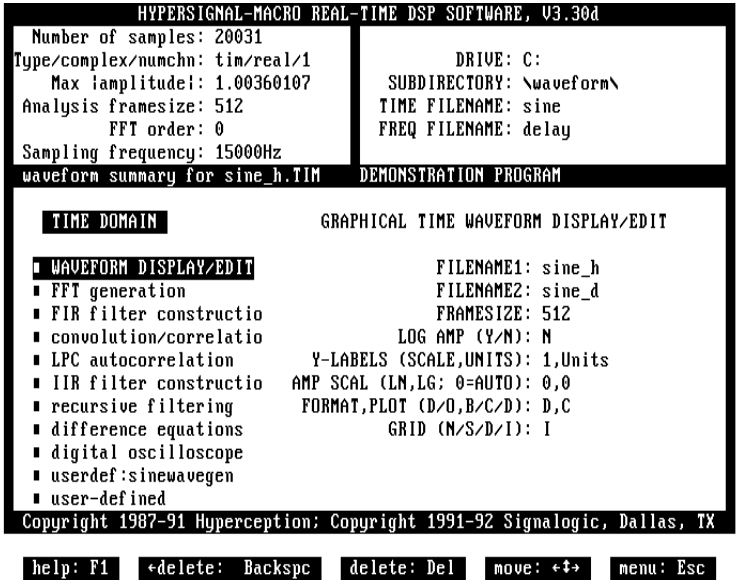


*Figure 14-5*

Ensure that the FORMAT subfield has been set to ‘D’, for dual-trace display. This will cause the Hilbert Transform-filtered sine wave to be displayed on the top trace, and the delay line-filtered sine wave to be displayed on the bottom trace. You should see results similar to Figure 14-6, below:

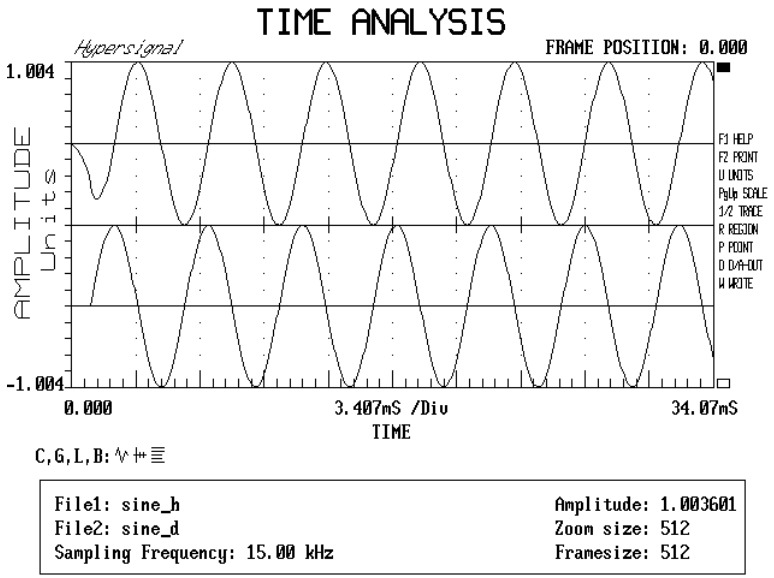


*Figure 14-6*

Note that the Hilbert Transform filter has done its job: the top sine wave has been delayed 90 degrees (1/4 period shift to right, or later in time) relative to the bottom wave. Note also the initial, inherent delay of each filter of 15.5 samples (half of the total impulse response length of 31 samples). If you press the 'T' key (random time-access), and then enter '1.03e-3' (15.5/15000 seconds), you will force the display to start immediately after the first 15.5 samples, causing the top trace to appear mathematically as a -cosine wave, and the bottom trace as a sine wave, as shown in Figure 14-7 below:

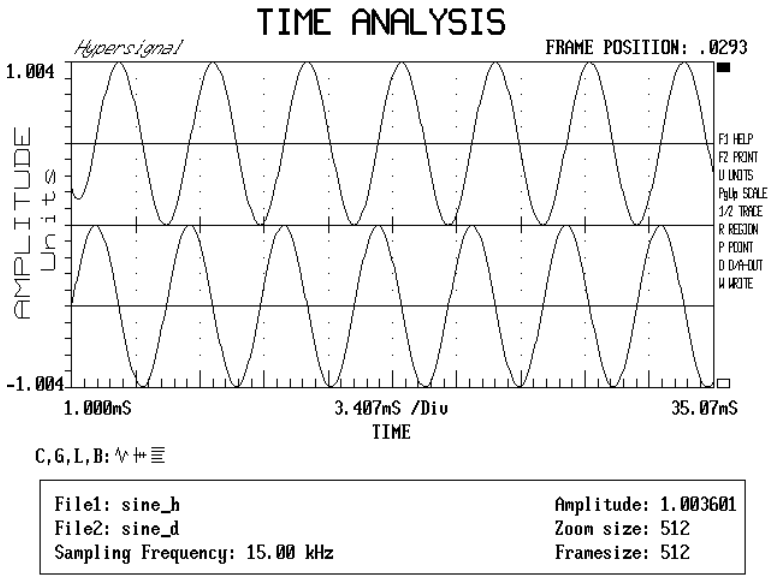


*Figure 14-7*

Use the arrow keys (or mouse motion) to move the target window around—the continuous readout at the bottom of the screen will allow you to make measurements based on where the target window is positioned. Note that pressing the 'U' key will toggle the current units display to different values, including straight sample number ("point") readout.

## 14.3 Digital Oscilloscope Verfication

Assuming you have a function generator hooked up to your current DSP/Analog board, the Digital Oscilloscope function represents a very quick way to verify digital filter operation. First, enter the following Digital Oscilloscope menu:

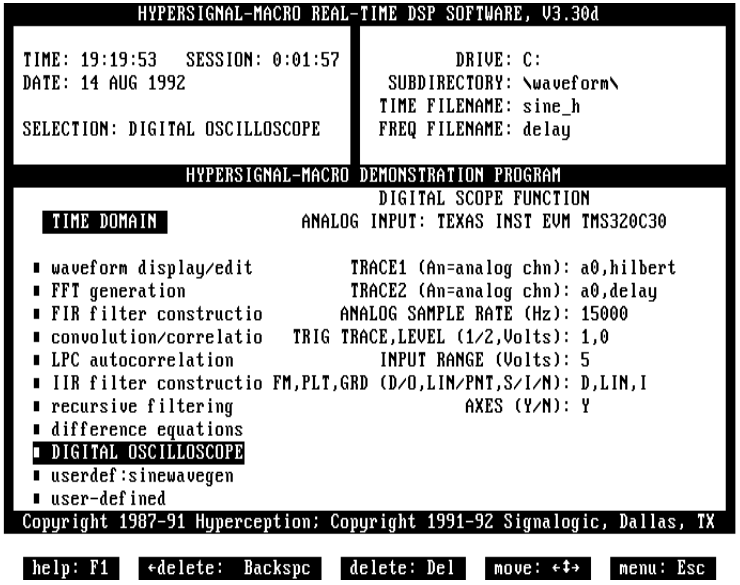


*Figure 14-8*

Second, after entering the last field and advancing to the display screen, tap the '+' key once or twice to activate a positive trigger level; this will stabilize the display in order to facilitate measurement. Ensure that your function generator isn't set to a frequency higher than 7500 Hz, otherwise you might be introducing aliasing error (the board's sample rate was only set for 15000 Hz). A rate of less than 1 kHz should provide an easily visible sine wave.

Third, press the right arrow key (or move the mouse to the right) until you can clearly see the input signals. By doing this you are changing the timebase value; the number of msec/div reads out just beneath the horizontal axis of the display.

If you have your function generator currently sending a sine wave to the channel 0 input on your DSP/Analog board, then you should see the trace 2 sine wave 90 degrees earlier in time (shifted 1/4 of sine wave period to left) than the trace 1 sine wave. Basically, you should see a -cosine wave on trace 1, and a sine wave on trace 2, as shown in Figure 14-9, below:

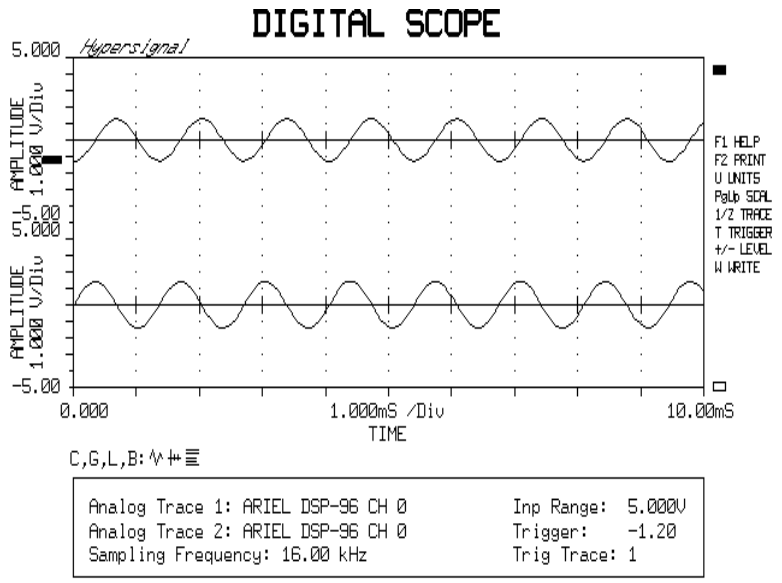


*Figure 14-9*

In the example above, the trigger level has been decreased (using the '-' key) to cause the trace 1 waveform to trigger near its minimum negative amplitude—see section 11.6, Timebase, Triggering, Vertical Scaling & Position, and Other Controls, for more detailed instructions on triggering. Also, the timebase controls (left and right arrow keys) were used to spread the trace display out in order to see several periods of the sine wave clearly. After the waveform has been adjusted and steadied (triggered) for viewing, try adjusting the frequency of the function generator output—note that the relative shift between the two trace displays stays constant, regardless of frequency!

*this page intentionally left blank*

# 15.0 MACRO EXAMPLES

The Hypersignal macro language can be described as a “watered-down combination of Pascal and C”. As with other languages, variables and programming constructs are available to store results, isolate often-used segments of code, invoke standard mathematical, file, and I/O functions, repeat segments of code, and make decisions based on results—in general, to solve problems. Unlike other languages, the macro language provides a direct “hot-line” to Hypersignal functions and displays, not only allowing them to be invoked and combined in any sequence, but to accept macro language variables as input parameters. Thus, high-level language problem solving becomes married to the powerful functions and displays in Hypersignal, making them appear almost as an extensive library of callable routines.

As with most high-level languages, variables must be “declared” to be one of several types before use, procedures can be defined before the “main body” of source code, mathematical operators and functions can be invoked, and various conditional and looping constructs are available. The macro language approach allows unique ability, such as gracefully transitioning from “interactive” mode to “automatic” mode. For example, the system can stop in the middle of an automated test procedure, wait for the operator to make measurements, save a display trace to disk file, set parameters, and then resume the test procedure—all without changing the current interactive function and its basic display screen.

This chapter discusses several practical examples of typical DSP processing using a macro language implementation. To achieve manually in Hypersignal, the examples considered would require multiple steps, as well as error-prone and distracting “thinking”: calculation of input parameters, and in some cases, what function sequence to use. In each example, a brief explanation of the macro and execution of the macro (including its input parameters) is discussed first, after which the macro is analyzed step-by-step.

For fully detailed information on the macro lanaguage, see Users Manual chapter 7, THE HYPERSIGNAL MACRO LANGUAGE. For a discussion of both the philosophical and practical aspects of creating macro language source files, see Users Manual section 4.10, Creating Macros.

### 15.0.1 Macro Source, Object, and Configuration Files

For convenience only, this section is presented to clarify the relationship between various macro files and filename extensions, and at what time, in what order, and how they get created. It is basically a repeat of Users Manual section 7.1, Macro Source, Object, and Configuration Files. This section is not necessary for understanding how the macro examples in the following sections are executed and why they work. In the framework of this tutorial, you can think of this section as one large POWER NOTE.

In order to create a macro, you must first create a “macro source file”. This file is then compiled by the built-in Hypersignal-Macro compiler, which in turn creates a “macro object file”. This object file is what Hypersignal actually executes. Thus, after a macro source file has been compiled once, only the macro object file need be accessed to execute the macro repeatedly. A “compiler mode” can be set in the Macro Manager menu that controls the circumstances under which compiling occurs. The compiler mode includes a “make mode” value which automatically compiles source files only if they have been changed. See Users Manual section 7.10, Compiling Macros, for more information.

Whenever you set up a macro for execution in the Macro Manager menu, a third file is automatically created: the “macro configuration file”. The configuration file basically stores in ASCII-text form all Macro Manager menu setup and input parameter fields. Also, the labels for the input parameter fields can be edited directly in this file, allowing you to customize the Macro Manager menu for different macros. This can be especially helpful for macros with several input parameters, or when you have several macros, or when running a macro after not using it for a long time. See Users Manual section 4.10.1.2, Parameter Passing From the Macro Manager Menu, for more information about the customization process.

Macro source files should normally have “.MAC” filename extensions; however, other extensions may be used by explicitly entering them in the Macro Manager menu MACRO FILENAME field when setting up the macro for compilation/execution.

Macro object files always have “.MXE” filename extensions—this extension is attached automatically by the built-in Hypersignal macro compiler.

Macro configuration files always have ".MCF" filename extensions—this extension is attached automatically by the Macro Manager menu function.

Macro source files are normally created in two ways:

1) by standard ASCII text-editor (manually)

2) by Hypersignal (automatically, using Macro Record function)

The first method is programming by any other name. You sit down with a text-editor, and, using the documentation and other macros as a guide, create the appropriate ".MAC" text file(s) to perform a certain function or series of functions. You may end up with a simple, short text file, or you may end up with a main text file and one or more "include" text files, possibly containing "procedures", that make up a more complicated macro.

The second method can be utilized by placing the Hypersignal software in "record mode", in which the software writes out all data field entries and interactive display commands as each function is executed from the main menu. This allows you to "step" through a sequence manually, and have the software generate the appropriate macro source file to then repeat, or "playback" this sequence later. See Users Manual section 4.10.2, Using the "Record" Mode to Create Macros, for more information.

Conceivably, it is also possible to generate macro source files in a third way: a separate, external program. This might be useful for creating a series of interrelated macros that must execute depending on external circumstances. An example of such an application might be a graphical interface intended for test system operators, who "build" a test suite by selecting icons; the application software they're using then creates the appropriate macros; finally, Hypersignal-Macro executes the test suite.

## 15.1 Sampling Rate Conversion

The following section explains the process of sampling rate conversion using Hypersignal-Macro. It includes a macro example that demonstrates the process, referring to the "samconv.mac" macro language file provided on distribution diskette #2. The macro takes an input waveform, multiplies its sampling rate by one integer (M), applies an interpolation filter, and then decimates the sampling rate by another integer (N). M and N must be chosen so that their ratio provides the correct scaling factor to

convert the input waveform sampling rate to the desired output waveform sampling rate.

The “samconv” macro accepts the following input parameters from the Macro Manager menu (note that a “samconv.mcf” macro configuration file, which contains the menu field labels for the macro, is also provided on distribution diskette #2):

1) input waveform name
2) M, integer by which to multiply input waveform sampling rate
3) N, integer by which to divide input waveform sampling rate
4) output waveform name

To execute “samconv”, first press the ‘F3’ key from the main menu to invoke the Macro Manager menu, and then enter the following fields:

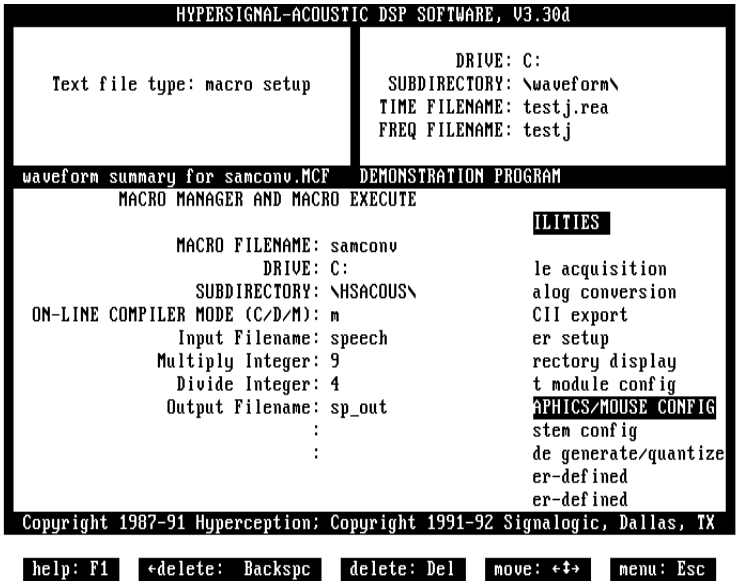


*Figure 15-1*

There are a few items to note about this menu. First, only the base filename (in this case, “samconv”) of the macro needs to be entered, since the program transparently uses different extensions to find the macro source file, menu configuration file, and create the actual executable file. If the source file has an extension other than “.MAC”, it must be entered explicitly in the MACRO FILENAME field. Second, the DRIVE and SUBDI-

RECTORY fields may well be different for your system, depending on its configuration and what subdirectory name you used when installing the software. Third, note that the COMPILER MODE field has been set to 'M', which means "make mode", and which instructs the on-line compiler to only compile the macro source file if changes have been made since the last compile. Finally, note the "customized" labels that the describe the usable input parameters to the macro. For any macro, the "filename.MCF" file, where filename is the base name of the macro, can be edited directly to set up labels. This concept allows the Macro Manager menu to function as a virtual menu, showing different labels and fields for different functions.

After entering the last field, or pressing <ctrl-Enter>, the macro will execute. If you have never run this macro before, you will see a message indicating that the macro is being compiled in the System Status area (upper left of main menu), otherwise you will see a brief message indicating the macro is being executed. During execution, you will see the various Hypersignal menu functions required by the macro flash by and execute. After execution completes, and control has been returned to the main menu, enter the following FFT Generation menus, allowing each one to execute:

```
                FILENAME: speech
               FRAMESIZE: 128
               FFT ORDER: 9
% FRAMESIZE OVERLAP: 50
WINDOW (R/Hm/Hn/Bl/Bt): Hm
REAL/IMAG OUTPUT (Y/N): N
          PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N
```

```
                FILENAME: sp_out
               FRAMESIZE: 288
               FFT ORDER: 9
% FRAMESIZE OVERLAP: 50
WINDOW (R/Hm/Hn/Bl/Bt): Hm
REAL/IMAG OUTPUT (Y/N): N
          PRECISION (I/R): R
PHASE/MAG APPROX (Y/N): N
```

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to 'none' entry). Note that we have entered identical FFT or-

der and window parameters, to allow side-by-side comparison in frequency domain displays.

**POWER NOTE: Note that the FRAMESIZE entry in the second menu above should be initialized by the program automatically, because the "samconv" macro will have pre-set this value in the "sp_out" waveform header, according to the 9/4 ratio sampling rate increase specified in the original Macro Manager menu. The same ratio also applies to both the length and framesize of the output waveform (in the case of framesize, 128*9/4, or 288).**

After both FFT Generation menus above have been executed, enter the following 2-D Spectrograph menu:

```
                    FILENAME1: sp_out
                    FILENAME2: speech
              POWER SPEC (P/M): M
             AMP SQUARE (Y/N): N
          LOG (N/+/-,N/base,Min): +
AMP SCALING (TOP,BOT,REF): 0,0,A
               COLOR (Cn/Gn/M): C2
                 SURFACE (C/R): C
      TIME SHOWN (S, 0=ALL): 0
```

You should see a display similar to Figure 15-2, below:

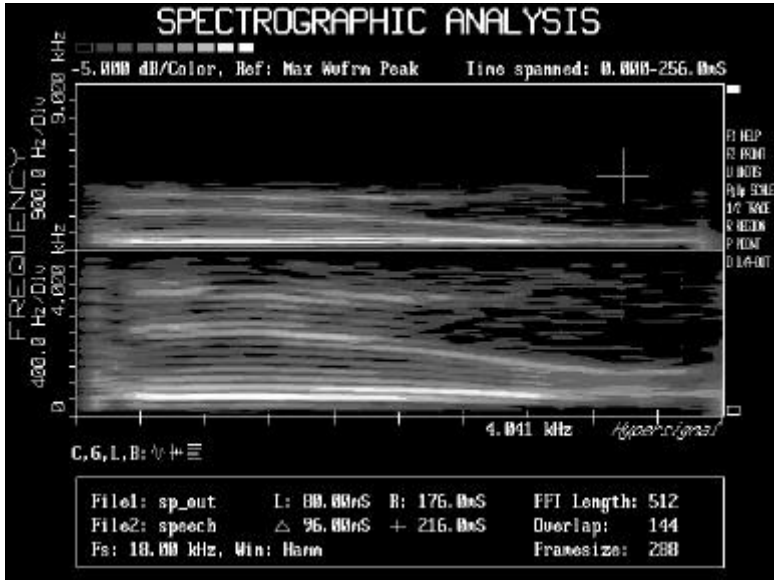


*Figure 15-2*

In this display, you can how the spectrum of the original “speech” waveform, shown on the bottom trace, has been compacted 2.25 times in the top trace, which has a sampling rate of 18 kHz. By ensuring that the time domain framesizes used in the FFT Generation functions for the waveforms are also in the 2.25 ratio, each FFT output frame in this display represents the same amount of time for both traces, allowing comparison viewing.

Now that we have executed the “samconv” macro, we will take a look at how it works. Below is the listing for the macro language source file:

```
; macro language text file for Hypersignal-Macro to perform sampling rate
;  conversion

; parameters passed to this macro from “Macro Manager” menu are:
; #1: input waveform name
; #2: M; integer amount to multiply previous sampling rate by
; #3: N; integer amount to divide previous sampling rate by
; #4: output waveform name

; Macro takes input waveform sampling rate and multiplies by M, then divides
; by N.  Thus, this macro forms a general-purpose sampling rate conversion
; “procedure” that can be used to either increase or decrease a given
; sampling rate.

; declare and calculate a few variables that will be needed

Integer: InSampFreq InLen InFrameSize InCutOffFreq
```

```
Integer: StretchSampFreq StretchLen StretchFrameSize
Integer: OutSampFreq OutLen OutFrameSize OutCutOffFreq
Integer: CutOffFreq D

InSampFreq = GetHeaderValue(#1, fs)  ; get values from input waveform header
InLen = GetHeaderValue(#1, len)
InFramesize = GetHeaderValue(#1, fr)
InCutOffFreq = InSampFreq div 2      ; “Nyquist” rate for input waveform

StretchSampFreq = InSampFreq * #2    ; calculate “stretched” values
StretchLen = InLen * #2
StretchFramesize = InFramesize * #2

OutSampFreq = StretchSampFreq div #3 ; calculate final converted values
OutLen = StretchLen div #3
OutFramesize = StretchFramesize div #3
OutCutOffFreq = OutSampFreq div 2    ; “Nyquist” rate for output waveform


; apply difference equation to stretch input waveform

DIFFERENCE EQUATIONS
   stretch                ; output waveform name
   y[n]=x1[n\#2]          ; #2 is second parameter passed to this macro
   blank
   blank
   blank
   #1                     ; input waveform
   none
   @StretchLen            ; length of input waveform * parameter #2
   @StretchSampFreq       ; sampling frequency of input waveform * parameter #2
   @StretchFramesize      ; framesize of input waveform * parameter #2
endfunc

; design FIR filter to interpolate/anti-alias between sample points of
; stretched waveform—use whichever cutoff frequency is less

if InCutOffFreq<OutCutOffFreq then
   CutOffFreq = InCutOffFreq
else
   CutOffFreq = OutCutOffFreq
endif

FIR FILTER CONSTRUCTION
   interp                 ; filter name
   kw                     ; design method (kw=Kaiser Window)
   @StretchSampFreq       ; sampling frequency (in Hz)
   0                      ; center frequency (in Hz)
   @CutOffFreq            ; bandwidth (in Hz)
   0                      ; left transition bandwidth (in Hz)
   250                    ; right transition bandwidth (in Hz)
   50                     ; stopband attenuation (in dB)
   auto                   ; let software choose number of taps
   1                      ; passband ripple (in dB)

   $RUN                   ; leave menu editor, execute function
```

```
   $C                    ; construct filter
endfunc

; now use convolution to apply the newly-created FIR interpolation filter

CONVOLUTION/CORRELATION
   stretch               ; input waveform name
   interp                ; filter name
   stretchf              ; output waveform name
   -                     ; '-' invokes convolution ('+' would be correlation)
endfunc

D = GetHeaderValue(interp, len)  ; get interp filter length

; use difference equation to "decimate" the filtered waveform by a factor of
; N (input parameter #3, above); ALSO, remove delay inserted by filter

DIFFERENCE EQUATIONS
   #4                    ; output waveform name, as specified in Macro Manager
   y[n]=x1[@D+n*#3]      ; parameter #3 is divide factor N (passed to macro)
   blank
   blank
   blank
   stretchf              ; input waveform "x1"
   none
   @OutLen               ; final length of output waveform
   @OutSampFreq          ; final sampling frequency of output waveform
   @OutFramesize         ; final framesize of output waveform
endfunc

; DONE.  New output waveform (name specified by parameter #4) is ready for
; subsequent processing.

DISPLAY(Macro Done, Menu is Active)
```

The "samconv" macro source file performs the following major steps:

1) Declares needed variables, and sets up intermediate "stretched" output waveform sampling rate, framesize, and length values by multiply the corresponding input waveform values by input parameter #2 (the "stretch" integer).

2) Applies a difference equation to create an intermediate, stretched waveform.

3) Designs and applies a lowpass filter to interpolate between samples of the stretched waveform. Note that the macro uses an if-then-else statement to determine the cutoff frequency of the lowpass filter—using half the final output sampling rate or half the stretch rate, whichever is less. If half the final output sampling rate is used, then the filter can be thought of

as an anti-aliasing filter; if half the stretch rate is used, the filter can be thought of as an interpolation filter.

4) Applies another difference equation to both decimate the stretched, filtered waveform by input parameter #3 (the “contract” integer), and remove delay inserted by the anti-alias/interpolation filter.

5) Note that the macro uses the Display command to indicate completion by displaying a message in the System Status area at the upper left of the main menu.

It would be a simple matter to incorporate the FFT Generation and frequency domain display steps presented earlier in this example directly into the “samconv” macro, and then invoke the $MANUAL interactive command to finish the macro inside a display, with manual controls active. However, in the spirit of reusable macros, the scope of the “samconv” macro is limited to only its prescribed function, allowing it be included in the form of an “include file” into other macros.

## 15.2 Frequency Zoom

The following section explains the process of frequency zoom using Hypersignal-Macro. It includes a macro example that demonstrates the process, referring to the “fzoom.mac” macro language file provided on distribution diskette #2. The macro takes an input waveform and shifts an arbitrary segment of bandwidth into the baseband (and correspondingly decreasing the overall sampling rate), allowing subsequent FFT processing to produce a high-resolution display.

The “fzoom” macro accepts the following input parameters from the Macro Manager menu (note that a “fzoom.mcf” macro configuration file is also provided on distribution diskette #2, which contains the menu field labels for the macro):

1) input waveform name
2) f1, beginning normalized frequency of arbitrary segment
3) f2, ending normalized frequency of arbitrary segment
4) output waveform name

To execute “fzoom”, first press the ‘F3’ key from the main menu to invoke the Macro Manager menu, and then enter the following fields:

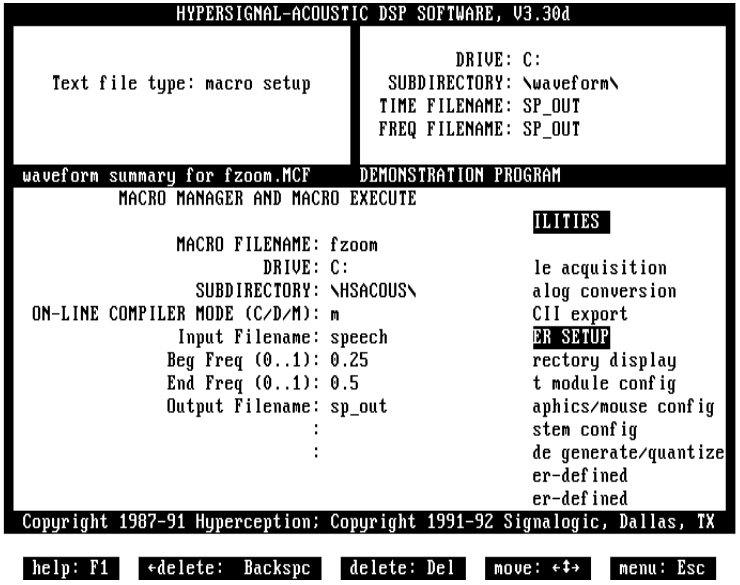


*Figure 15-3*

There are a few items to note about this menu. First, only the base filename (in this case, “fzoom”) of the macro needs to be entered, since the program transparently uses different extensions to find the macro source file, menu configuration file, and create the actual executable file. If the source file has an extension other than “.MAC”, it must be entered explicitly in the MACRO FILENAME field. Second, the DRIVE and SUBDIRECTORY fields may well be different for your system, depending on its configuration and what subdirectory name you used when installing the software. Third, note that the COMPILER MODE field has been set to ‘M’, which means “make mode”, and which instructs the on-line compiler to only compile the macro source file if changes have been made since the last compile. Finally, note the “customized” labels that the describe the usable input parameters to the macro. For any macro, the “filename.MCF” file, where filename is the base name of the macro, can be edited directly to set up labels. This concept allows the Macro Manager menu to function as a virtual menu, showing different labels and fields for different functions.

After entering the last field, or pressing the <ctrl-Enter> key sequence, the macro will execute. If you have never run this macro before, you will see a message indicating that the macro is being compiled in the System Status

area (upper left of main menu), otherwise you will see a brief message indicating the macro is being executed. During execution, you will see the various Hypersignal menu functions required by the macro flash by and execute. After execution completes, and control has been returned to the main menu, enter the following FFT Generation menus, allowing each one to execute:

| | |
|---|---|
| FILENAME: | speech |
| FRAMESIZE: | 128 |
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Hm |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

| | |
|---|---|
| FILENAME: | sp_out |
| FRAMESIZE: | 32 |
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Hm |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields will only appear if you are using a non-real-time version of software, or the DSP ACCELERATOR field in the System Configuration menu has been disabled (set to 'none' entry). Note that we have entered identical FFT order and window parameters, to allow side-by-side comparison in frequency domain displays.

**POWER NOTE:** **Note that the FRAMESIZE entry in the second menu above should be initialized by the program automatically, because the "fzoom" macro will have pre-set this value in the "sp_out" waveform header, according to the 1/4 ratio sampling rate decrease implied by choosing a normalized segment of bandwidth upon which to zoom starting at 0.25 and ending at 0.5 in the original Macro Manager menu. The same ratio also applies to both the length and framesize of the output waveform (in the case of framesize, 128∗1/4, or 32).**

After both FFT Generation menus above have been executed, enter the following 2-D Spectrograph menu:

```
                    FILENAME1: sp_out
                    FILENAME2: speech
                POWER SPEC (P/M): M
               AMP SQUARE (Y/N): N
            LOG (N/+/-,N/base,Min): +
AMP SCALING (TOP,BOT,REF): 0,0,A
               COLOR (Cn/Gn/M): C2
                  SURFACE (C/R): C
          TIME SHOWN (S, 0=ALL): 0
```

You should see a display similar to Figure 15-4, below:

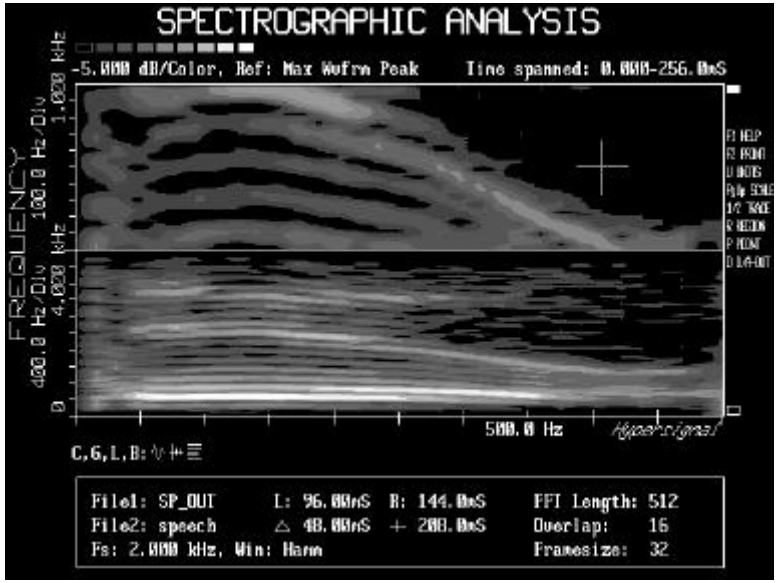


*Figure 15-4*

In this display, you can how the 25% to 50% segment of the original “speech” waveform spectrum, shown on the bottom trace, has been zoomed to full display proportion, and now has an overall effective sampling rate of 2 kHz. By ensuring that the time domain framesizes used in the FFT Generation functions for the waveforms are also in the 1/4 ratio, each FFT output frame in this display represents the same amount of time for both traces, allowing comparison viewing.

Now that we have executed the "fzoom" macro, we will take a look at how it works. Below is the listing for the macro language source file:

```
; macro language text file for Hypersignal-Macro to perform frequency zoom

; parameters passed to this macro from "Macro Manager" menu are:
; #1: input waveform name
; #2: normalized beginning freqency of zoom segment
; #3: normalized end freqency of zoom segment
; #4: output waveform name

; NOTES: Normalized frequencies range from 0 to 1, where 1 is equivalent
; to half the sampling rate.  F1 and f2 should be entered so that, when
; multiplied by half the sampling rate, more or less an integer results.

; Macro takes input waveform, spectrally inverts it, lowpass filters unwanted
; lower bandwidth (which is now upper bandwidth after inversion), and then
; multiplies by cos(2pinf/Fs), where f is the necessary shift frequency to
; reflect the target segment back into the baseband, non-inverted.  Finally,
; a lowpass filter-decimator stage is applied to remove unwanted upper
; bandwidth and reduce overall sampling rate.

; declare and calculate a few variables that will be needed

integer InSampFreq InLen InFrameSize
integer OutSampFreq OutLen OutFrameSize
integer Nyquist deci delay
real f1,f2,rtemp,CutOffFreq

InSampFreq = GetHeaderValue(#1, fs)  ; get values from input waveform header
InLen = GetHeaderValue(#1, len)
InFramesize = GetHeaderValue(#1, fr)

Nyquist=InSampFreq div 2

f1=#2*Nyquist
f2=#3*Nyquist

; apply difference equation to spectrally invert waveform

DIFFERENCE EQUATIONS
   invert                ; intermediate output waveform name
   y[n]=x1[n]*(-1)^n     ; inversion
   blank
   blank
   blank
   #1                    ; input waveform
   none                  ; length, Fs, and Fr of output same as input
   0
   0
   0
endfunc

; design FIR filter to zap unwanted lower bandwidth

CutOffFreq=Nyquist-f1

FIR FILTER CONSTRUCTION
```

```
    zlower              ; filter name
    kw                  ; design method (kw=Kaiser Window)
    @InSampFreq         ; sampling frequency (in Hz)
    0                   ; center frequency (in Hz)
    @CutOffFreq         ; bandwidth (in Hz)
    0                   ; left transition bandwidth (in Hz)
    150                 ; right transition bandwidth (in Hz)
    50                  ; stopband attenuation (in dB)
    auto                ; let software choose number of taps
    1                   ; passband ripple (in dB)

    $RUN                ; leave menu editor, execute function
    $C                  ; construct filter
endfunc

; apply FIR filter

CONVOLUTION/CORRELATION
    invert              ; input waveform name
    zlower              ; filter name
    inv_fil             ; output waveform name
    -                   ; '-' invokes convolution ('+' would be correlation)
    no_change           ; EST doesn't matter, since filter was created by
                        ; built-in design function
endfunc

delay = GetHeaderValue(zlower, len) ; get filter length

; apply difference equation to reflect waveform frequency about 0 (dc) and
; also remove filter delay

DIFFERENCE EQUATIONS
    het                 ; intermediate output waveform name
    cos(2*pi*n*         ; heterodyne equation
    @CutOffFreq/Fs)*
    x1[n+@delay/2]
    blank
    inv_fil             ; input waveform
    none                ; length, Fs, and Fr of output same as input
    0
    0
    0
endfunc

; design FIR filter to zap unwanted upper bandwidth

CutOffFreq=f2-f1

FIR FILTER CONSTRUCTION
    zupper              ; filter name
    kw                  ; design method (kw=Kaiser Window)
    @InSampFreq         ; sampling frequency (in Hz)
    0                   ; center frequency (in Hz)
    @CutOffFreq         ; bandwidth (in Hz)
```

```
    0                       ; left transition bandwidth (in Hz)
    150                     ; right transition bandwidth (in Hz)
    50                      ; stopband attenuation (in dB)
    auto                    ; let software choose number of taps
    1                       ; passband ripple (in dB)

    $RUN                    ; leave menu editor, execute function
    $C                      ; construct filter
endfunc

; apply FIR filter

CONVOLUTION/CORRELATION
    het                     ; input waveform name
    zupper                  ; filter name
    het_fil                 ; output waveform name
    -                       ; '-' invokes convolution ('+' would be correlation)
    no_change               ; EST doesn't matter, since filter was created by
                            ; built-in design function
endfunc

delay = GetHeaderValue(zupper, len) ; get filter length

; last step:  difference equation to "decimate" the filtered waveform by the
; ratio of the original bandwidth to the bandwidth of the zoom segment
; ALSO: eliminate delay inserted by most recent filter

rtemp=Nyquist/CutOffFreq
deci=ROUND(rtemp)
OutLen=InLen div deci
OutSampFreq=InSampFreq div deci
OutFramesize=InFramesize div deci

DIFFERENCE EQUATIONS
    #4                      ; output filename, as specified in Macro Manager menu
    x1[n*@deci+@delay/2]    ; deci contains integer decimation factor
    blank
    blank
    blank
    het_fil                 ; input waveform "x1"
    none
    @OutLen                 ; final length of output waveform
    @OutSampFreq            ; final sampling frequency of output waveform
    @OutFramesize           ; final framesize of output waveform
endfunc

; DONE.  New output waveform (name specified by parameter #4) is ready for
; subsequent FFT processing, which will concentrate FFT resolution over
; reduced bandwidth segment.

DISPLAY(Macro Done, Menu is Active)
```

The "fzoom" macro source file performs the following major steps:

1) Declares needed variables, and uses the normalized zoom segment input parameters (ranging from 0 to 1) to establish actual target sampling frequency values based on the sampling rate value in the input waveform header.

2) Applies a difference equation to spectrally invert the input waveform, creating a waveform where frequencies at dc and the Nyquist rate (and in between) are swapped.

3) Designs and applies a lowpass filter to eliminate unwanted lower bandwidth (which is now upper bandwidth because of the spectral inversion).

4) Applies another difference equation to both partially re-invert, or reflect the desired segment about dc, and remove delay inserted by the previous filter. The equation achieves the reflection by multiplying the waveform by a cosine, which causes heterodyning, producing copies of the spectrum at ∫ the cosine waveform frequencies.

5) Designs and applies another lowpass filter to eliminate the positive copy of the spectrum after the shift (may be considered an anti-aliasing filter).

6) Applies a final difference equation to decimate the output waveform to the final, overall sampling rate, and also remove delay inserted by the previous filter.

7) Note that the macro uses the Display command to indicate completion by displaying a message in the System Status area at the upper left of the main menu.

It would be a simple matter to incorporate the FFT Generation and frequency domain display steps presented earlier in this example directly into the “fzoom” macro, and then invoke the $MANUAL interactive command to finish the macro inside a display, with manual controls active. However, in the spirit of reusable macros, the scope of the “fzoom” macro is limited to only its prescribed function, allowing it be included in the form of an “include file” into other macros.

## 15.3 Analog Conversion Macros

The following short macro examples demonstrate how to invoke the Analog Conversion function from a macro language file. *These examples are*

*valid for real-time software versions only.* The macro listings shown below require no input parameters.

```
; example macro language file that invokes Analog Conversion menu selection, acquires
some data, and returns to main menu

ANALOG CONVERSION
   example              ; example input waveform name
   a                    ; a=acquire, am=acquire w/monitor, g=generate
   0                    ; channel 0 only
   1,1                  ; input, output gain = 1
   0                    ; trigger channel is 0
   INT,0,0              ; internal trigger, level=0 (no trigger), delay=0
   20000                ; number of samples
   12000                ; sampling rate (in Hz)
   0                    ; framesize (used only in post-processing functions
                        ; 0 prevents auto-initialization in other functions)
   0,0                  ; input, output offset

   $RUN                 ; run
   $Y                   ; answer "BEGIN ... (Y/N)" prompt
   $ESC                 ; return to main menu when done
endfunc
```

The important aspect to note about the macro listing above is how the "BEGIN ACQUISITION (Y/N)?" prompt is handled. In most cases, user prompts are disabled in macro mode, the basic assumption being that the user needs things to get done, and wouldn't be using macro mode if he/she wasn't prepared to ignore the usual prompts regarding such actions as overwriting an output waveform file that already exists. However, in the case of Analog Conversion, it may be necessary to synchronize the actual start of continuous input/output with some other event; hence, the start prompt is still active. In the listing above, the prompt is answered by using an interactive command. Anytime a reserved command does not follow the '$' interactive command symbol, the text is fed character by character to the program, as if it was being typed at the keyboard. Thus, the '$Y' interactive command is interpreted by the prompt as a 'Y'. (This same technique can also be used for other prompts in the program that expect arbitrary input, such as a filename prompt inside a waveform display.)

Note that the Analog Conversion function can be run more than once without returning to the main menu, say for example, after waiting for operator keyboard entry, by placing more interactive commands before the '$ESC' command, as in the following listing:

```
ANALOG CONVERSION
```

```
   example              ; example input waveform name
   a                    ; a=acquire, am=acquire w/monitor, g=generate
   0                    ; channel 0 only
   1,1                  ; input, output gain = 1
   0                    ; trigger channel is 0
   INT,0,0              ; internal trigger, level=0 (no trigger), delay=0
   20000                ; number of samples
   12000                ; sampling rate (in Hz)
   0                    ; framesize (used only in post-processing functions
                        ; 0 prevents auto-initialization in other functions)
   0,0                  ; input, output offset

   $RUN                 ; run
   $Y                   ; answer "BEGIN ... (Y/N)" prompt
   $DELAY(2000)         ; delay 2 seconds (after conversion has completed)
   $WAITFORKEY(C)       ; wait for operator to enter 'continue' key
   $RUN                 ; run again
   $ESC                 ; return to main menu when done
endfunc
```

## 15.4 Stimulus & Response Macro

The following macro example demonstrates the process of stimulus & response and subsequent analysis, and requires the "stim.mac" macro language file provided on distribution diskette #2 (real-time software versions only). The "stim.mac" macro creates a sweep waveform, uses the Analog Conversion function to perform a stimulus & response operation, uses the cross-correlation function to extract an impulse response, uses the FFT Generation function to calculate a frequency response, and displays the result in the Magnitude Display function, leaving control with the user inside the display.

The "stim" macro accepts three input parameters from the Macro Manager menu: sampling frequency (in Hz), stimulus length (in samples), and maximum system delay (in samples; total of stimulus length and maximum system delay is limited to 16384). The sampling frequency and stimulus length together effectively control the frequency resolution of the resulting transfer function (calculated as sampling rate divided by stimulus length). Note that a "stim.mcf" macro configuration file, which contains the menu field labels for the macro, is also provided on distribution diskette #2.

To execute "stim.mac", first press the 'F3' key from the main menu to invoke the Macro Manager menu, enter 'stim' in the MACRO FILENAME field, and enter the desired sampling rate and stimulus length. For maxi-

mum system delay, enter a safe number of samples (an amount of time equal to this number divided by the sampling rate). However, you should not make the maximum delay value any larger than necessary, because this could significantly and unnecessarily slow post-processing after the stimulus & response operation completes. After the input parameters are entered, press the <ctrl-Enter> key sequence. Make sure that you have the input and output to the system under test connected to the current DSP/Acquisition hardware channel 0 input and channel 0 output, respectively. *You should double-check the Analog Conversion menu inside the "stim" macro to make sure gain, trigger, and other fields are correct for your hardware*, as often times these fields are different for specific hardware. Also, you should have the DSP ACCELERATOR field inside the System Configuration menu set, as otherwise the cross-correlation invoked by the "stim" macro will take an inordinately long time to execute. Below is the listing of the macro language source file for "stim.mac":

```
; stim.mac:  example macro language file that performs stimulus & response
; operation, starting with stimulus waveform creation and ending with
; frequency response display

; input parameters:
; #1:  sampling frequency (in Hz)
; #2:  stimulus length (in samples; freq resolution = samp freq / length)
; #3:  maximum delay length (in samples)

; create linear sweep waveform

integer total_len
total_len = #2 + #3

DIFFERENCE EQUATIONS
   sweep
   1500*sin(pi*n*n/2/        ; sweep equation
   (#2-1))*
   u[#2-n]
   blank
   none
   none
   @total_len                ; length = stimulus length + delay length
   #1                        ; sampling rate (input parameter #1)
   @total_len                ; framesize same as length
endfunc

; perform stimulus & response

ANALOG CONVERSION
   test                      ; input waveform name
   a,sweep                   ; a=acquire, am=acquire w/monitor, g=generate
   0                         ; channel 0
   1,1                       ; input, output gain = 1 (may differ depending on hw)
   0                         ; trigger channel is 0
```

```
    INT,0,0                 ; internal trigger, level=0 (no trigger), delay=0
    @total_len              ; number of samples (total length)
    #1                      ; sampling rate (input parameter #1)
    @total_len              ; post-acquisition framesize:  same as length
    0,0                     ; input, output offset = 0

    $RUN                    ; run
    $Y                      ; answer “BEGIN ... (Y/N)” prompt
    $ESC                    ; return to main menu
endfunc

; cross-correlate stimulus output with sweep to extract impulse response

CONVOLUTION/CORRELATION
    test                    ; input waveform name
    sweep                   ; filter name
    impulse                 ; output waveform name
    +                       ; ‘+’ invokes cross-correlation
endfunc

; set type attribute for impulse response to “H[n]” directly in waveform header

integer type_attribute   ; declare variable to hold type attribute

type_attribute = GetHeaderValue(impulse, TypeData)
type_attribute = type_attribute | 2  ; set data type field to H[n]
PutHeaderValue(impulse, Typedata, type_attribute)

; calculate frequency response

FFT GENERATION
    impulse                 ; input filename
    4096                    ; analysis framesize = 4096 samples
    12                      ; FFT length = 4096
    0                       ; no overlap
    R                       ; rectangular window
    N                       ; no real/imaginary output
    R                       ; use real precision (this field only appears if
                            ; no DSP Accelerator installed)
    N                       ; disable phase/mag approx. (this field only
                            ; appears if no DSP Accelerator installed)
endfunc

; display frequency response, leave display in manual mode

MAGNITUDE DISPLAY
    impulse                 ; input filename
    blank
    M                       ; magnitude format
    N                       ; no amplitude-squared
    +,N                     ; log amplitude, linear frequency
    1,no_units              ; scale factor, units
    0,0,S                   ; use system dB reference for vertical plot limits
    D,C                     ; split-screen if dual-trace, connected sample drawing
    I                       ; intersected grid
```

```
   $RUN                ; leave menu editor, begin display
   $MANUAL             ; return control to user
endfunc
```

There are some important notes about execution of the “stim” macro language file, as shown above. First, note that samples occuring after the entered value of stimulus length but before the sum of the entered values of stimulus length and maximum delay are zeroed, to allow for system delay. You should avoid choosing a stimulus length and maximum delay such their sum exceeds 16384 (or an amount of time equal to 16384 divided by the sampling rate). Second, the FFT length in the macro has been set to 4096, the maximum limit of the FFT Generation function. If the impulse response waveform, created by the cross-correlation function, has significant energy past 4096 samples, then the final frequency domain result may not be accurate. Third, note that the FFT length is independent of the stimulus length; thus, even though 8 FFT frames are generated because of the long stimulus being used as cross-correlation input, only the first FFT frame is valid. *This implies that Power Spectra Generation should not be performed on the output frequency domain “impulse” waveform.* Finally, note that the macro directly sets the type attribute in the “impulse” waveform header to indicate transfer function data, or “H[n]”. Note the use of the logical OR to accomplish this (using the ‘|’ symbol).

The “stim” macro source file performs the following major steps:

1) Creates a linear sweep sine wave with length specified by input parameter #2 for use as a stimulus. Zeros may be added after the stimulus length to allow for system delay, depending on the value of input parameter #3 (maximum system delay). The frequency resolution of the stimulus is thus the sampling rate divided by the stimulus length. The time resolution (for example, the amount of time to which a system delay can be resolved) is 1 over the sampling rate.

2) Invokes the Analog Conversion function to perform stimulus & response measurement. Note that the DSP/Analog board used must have at least 16384 samples of memory available for stimulus output. Consult the Hardware Reference Guide for specific hardware under the “Onboard Memory Usage” sections for memory availability information.

3) Uses the Convolution/Correlation function to cross-correlate the system response with the stimulus, producing the system impulse response.

4) Uses the FFT Generation function to calculate the system frequency response from the system impulse response. Note that the 4096 pt. limitation in the FFT Generation function means that the maximum possible frequency resolution for display is the sampling rate divided by 4096.

5) Displays the frequency response in the Magnitude Display function, returning control to manual mode inside this display.

See section 12.3, Time Domain Analysis, inside chapter 12, STIMULUS & RESPONSE, for detailed information and a step-by-step manual mode example of the above process.

## 15.5 Interactive Display Macros

The following subsections contain several macro examples of invoking and controlling interactive displays in Hypersignal, including instrumentation displays, from macro language files. The examples concentrate on the use of interactive commands inside displays, invoking and answering prompts, returning to the data-entry menu to change values, and controlling the amount and position of data displayed. For a deluxe version of interactive waveform control, including delays and zooms, consult the "demo.mac" macro source file provided on distribution diskette #2. This macro is provided with Hypersignal demo diskettes to demonstrate the interactive displays as if actually being used manually by an engineer or analyst.

### 15.5.1 Waveform Display/Edit Cut & Paste Macro

The following macro example demonstrates a cut-and-paste operation from the Waveform Display/Edit function, and is based on the "cut.mac" macro language file provided on distribution diskette #2. The "cut.mac" macro cuts, or removes, a section of waveform, and then re-inserts it at a different location. A "point mode" of editing is also available in the Waveform Display/Edit function (enabled by the 'P' key), in which free-form amplitude changes can be made to the waveform using the mouse, but the "cut.mac" text below does not invoke this mode. The macro returns control to the user inside the Waveform Display/Edit function, with the editing changes visible. After control returns to manual, if you choose to experiment with point mode, you can use the 'R' key to return to cut-and-paste (region) mode.

**NOTE: If editing changes have been made to a waveform(s) in the Waveform Display/Edit function, then upon returning to the main menu from the data-entry menu, or upon attempting to display a new waveform, the software will prompt you whether to keep changes to the current waveform(s). If you answer yes, the original waveform(s) will be renamed to a "backup" file and the current changes will be made "final". If you answer no, the changes will be discarded. At this point the software will let you return to the main menu or display new waveforms. This applies to editing changes made in either the cut-and-paste or point modes. Thus, after running the "cut.mac" macro below, you will see this prompt before returning to the main menu. Since the "cut.mac" macro is just an example, you will probably choose to discard the changes. For a detailed discussion of waveform editing, see Users Manual section 4.3.4, Editing Waveforms.**

To execute "cut.mac", first press the 'F3' key from the main menu to invoke the Macro Manager menu, and enter 'cut' in the MACRO FILENAME field. Then press the <ctrl-Enter> key sequence. Below is the listing of the macro language source file for "cut.mac":

```
; macro language text file to demonstrate cut-and-paste from inside Waveform
; Display/Edit function

Waveform Display/Edit
   speech              ; input waveform name
   blank               ; dual-waveform display
   1000                ; set display framesize to control size of cut segment
   N                   ; no log-amplitude
   1,AD_Units          ; no scale-factor, untis
   0                   ; autoscale
   D,C                 ; split-screen display, connected sample drawing
   I                   ; intersected grid

   $RUN                ; run

   $T                  ; invoke random-access prompt
   $0.1875 $ENTER      ; enter random-access time value (in seconds)

   $DEL $D             ; cut entire display (framesize samples)

   $T $0 $ENTER        ; re-position display to start of waveform
   $INS $I             ; insert cut segment at start of waveform

   $ESC                ; return to data-entry menu, double framesize
```

```
   $REPEAT(5) $UP
   $2000                ; set framesize field

   $RUN                 ; go back into display to see changes

; use $MANUAL to stay in display, observe edit changes, continue processing
; after manual Esc key


   $MANUAL

; remove $MANUAL, use following sequence to discard changes, continue
; processing automatically

;  $ESC                ; return to data-entry menu
;  $ESC                ; return to main menu
;  $N                  ; decline to keep changes (this example only)

; remove $MANUAL, use following sequence to keep changes, continue
; processing automatically

;  $ESC                ; return to data-entry menu
;  $ESC                ; return to main menu
;  $Y                  ; keep changes (this example only)
endfunc
```

There are some interesting notes about execution of the “cut” macro language file, as shown above. First, the macro uses the “general” form of the interactive command to invoke and answer a software prompt. The general form includes all text or words not reserved; i.e. not listed as interactive commands in Users Manual section 7.3.2, Issuing Interactive Commands From Macros. In the example above, the ‘$T’ command is used to invoke a random-access time prompt, and then a ‘$0.1875’ command is used to enter a general numeric value. Note that after the numeric entry, a ‘$ENTER’ command is used to indicate the end of the numeric entry and cause the program to complete the prompt and act upon the entry. Second, the macro uses interactive commands to return to the data-entry menu, change a menu field, and re-enter the display. This approach can be useful when changing values specified by the initial data-entry menu field values prior to the start of interactive commands becomes necessary. Third, the macro returns control to manual mode while still inside the display. If the macro contained more processing after the “endfunc” statement, then this processing would resume after you returned to the main menu. If no processing is specified after the “endfunc” statement, then for all intents and purposes, the macro is finished. Note that you can stay inside the display for as long as desired and perform any amount of measurement and other operations, including return to the data-entry menu as

many times as necessary; the macro will only resume upon exit to the main menu.

**NOTE: In the macro text above, interactive commands are used to alter and enter some data-entry fields—but not waveform filename fields. During normal operation, Hypersignal-Macro initializes various fields in data-entry menus automatically based on information contained in waveform headers. This usually happens when a data-entry menu is first entered, requiring the program to "load" a waveform specified in a waveform filename field. It can also happen when you change waveform filename field entries inside a data-entry menu. This type of operation poses a basic conflict with macro operation, since often times data-entry menu fields are specified in the macro. When a function or display is invoked from a macro, all data-entry fields not marked 'no_change' or 'don't_care' take precedence over the automatic load and initialize. *However*, if, from inside the macro file, you use interactive commands to cause another waveform load (by causing a filename entry field to be "entered"), then the automatic load will take precedence, and one or more data-entry fields containing previous macro file inputs may be overwritten. Using interactive commands to enter waveform filenames must be done carefully and is not usually recommended.**

The "cut" macro source file performs the following major steps:

1) First repositions the start of the display to 0.1875 sec using the random-access time prompt ('$T' interactive command). Since the initial framesize was specified as 1000 samples, the display will then show 0.125 sec of data starting at 0.1875 sec. The combination of FRAMESIZE field entry and the random-access time prompt can be used to exactly position a desired segment of waveform inside the current display frame.

2) Cuts the segment of the waveform currently displayed using the '$DEL' (delete) interactive command. Note that a '$D' command must also be entered after the delete command: this is because the software prompts for either a delete ('$D') operation or a copy ('$C') operation. You can experiment with this easily in manual mode. Cut-and-paste (delete/copy and insert/write) operations in manual mode work either on the

entire display frame or, if displayed, on the current target window, which is controlled by the mouse. However, in macro mode, it is advisable to not use the target window, since there is no way to position the window exactly using macro interactive commands (you could use the $RIGHT, $LEFT, $UP, and $DOWN interactive commnds, but positioning would still be inexact).

3) Again uses the random-access time prompt to resposition the display to another segment of data.

4) Inserts the previously cut segment of waveform at the new position using the '$INS' (insert) interactive command. Note that a '$I' command must also be entered after the insert command: this is because the software prompts for either an insert ('$I') operation or an overwrite ('$W') operation. You can experiment with this easily in manual mode. The segment is inserted at the left edge of the display frame (if the target window were displayed, then the insert would occur at the left edge of the target window).

5) Returns to the data-entry menu, re-enters the FRAMESIZE field, and returns to the display again (to better show the editing changes). At this point, 2000 samples (0.25 sec) of data is displayed starting at 0 sec.

6) Returns control to manual mode using the '$MANUAL' interactive command.

See Users Manual section 4.3.4, Editing Waveforms, for a fully detailed discussion of the cut-and-paste and point modes of waveform editing.

## 15.5.2 Waveform Display/Edit Write to File Macro

The following macro example demonstrates a write-to-file operation from the Waveform Display/Edit function, and is based on the "write.mac" macro language file provided on distribution diskette #2. The "write.mac" macro writes out a segment of waveform data to a waveform file named "chunk" and then displays the new waveform.

To execute "write.mac", first press the 'F3' key from the main menu to invoke the Macro Manager menu, and enter 'write' in the MACRO FILENAME field (the macro requires no input parameters). Then press the <ctrl-Enter> key sequence. Below is the listing of the macro language source file for "write.mac":

```
; macro language text file to demonstrate write-to-file from inside Waveform
; Display/Edit function

Waveform Display/Edit
   speech                ; input waveform name
   blank                 ; dual-waveform display
   1000                  ; set display framesize to control size of write
   N                     ; no log-amplitude
   1,AD_Units            ; no scale-factor, untis
   0                     ; autoscale
   D,C                   ; split-screen display, connected sample drawing
   I                     ; intersected grid

   $RUN                  ; run

   $T                    ; invoke random-access prompt
   $0.1875 $ENTER        ; enter random-access time value (in seconds)

   $W                    ; invoke write-to-file prompt
   $chunk $ENTER         ; enter name of segment, write entire display (framesize
                         ; samples) to new waveform file

   $ESC                  ; return to data-entry menu, double framesize
   $REPEAT(7) $UP
   $chunk $ENTER         ; set filename1 field to segment just written

   $DOWN                 ; reset framesize, because load of "chunk" has re-
   $1000                 ; initialized it

   $RUN                  ; go back into display to see changes

; use $MANUAL to stay in display, continue processing after manual Esc key

   $MANUAL

; remove $MANUAL, use following sequence to return to main menu, continue
; processing automatically

;  $ESC                  ; return to data-entry menu
;  $ESC                  ; return to main menu
endfunc
```

There are some interesting notes about execution of the "write" macro language file, as shown above. First, the macro uses the "general" form of the interactive command to invoke and answer a software prompt. The general form includes all text or words not reserved; i.e. not listed as interactive commands in Users Manual section 7.3.2, Issuing Interactive Commands From Macros. In the example above, the '$W' command is used to invoke the write-to-file prompt, and then a '$chunk' command is used to enter a general string value. Note that after the string entry, a '$ENTER'

command is used to indicate the end of the string entry and cause the program to complete the prompt and act upon the entry. Second, the macro uses interactive commands to return to the data-entry menu, change a menu field, and re-enter the display. This approach can be useful when changing values specified by the initial data-entry menu field values prior to the start of interactive commands becomes necessary. Third, the macro returns control to manual mode while still inside the display. If the macro contained more processing after the "endfunc" statement, then this processing would resume after you returned to the main menu. If no processing is specified after the "endfunc" statement, then for all intents and purposes, the macro is finished. Note that you can stay inside the display for as long as desired and perform any amount of measurement and other operations, including return to the data-entry menu as many times as necessary; the macro will only resume upon exit to the main menu.

**NOTE: In the macro text above, interactive commands are used to alter and enter some data-entry fields, including a waveform filename field. During normal operation, Hypersignal-Macro initializes various fields in data-entry menus automatically based on information contained in waveform headers. This usually happens when a data-entry menu is first entered, requiring the program to "load" a waveform specified in a waveform filename field. It can also happen when you change waveform filename field entries inside a data-entry menu. This type of operation poses a basic conflict with macro operation, since often times data-entry menu fields are specified in the macro. When a function or display is invoked from a macro, all data-entry fields not marked 'no_change' or 'don't_care' take precedence over the automatic load and initialize. *However,* if, from inside the macro file, you use interactive commands to cause another waveform load (by causing a filename entry field to be "entered"), then the automatic load will take precedence, and one or more data-entry fields containing previous macro file inputs may be overwritten. Using interactive commands to enter waveform filenames must be done carefully and is not usually recommended, although shown in the example above.**

The "write" macro source file performs the following major steps:

1) First repositions the start of the display to 0.1875 sec using the random-access time prompt ('$T' interactive command). Since the initial framesize was specified as 1000 samples, the display will then show 0.125 sec of data starting at 0.1875 sec. The combination of FRAMESIZE field entry and the random-access time prompt can be used to exactly position a desired segment of waveform inside the current display.

2) Writes the segment of the waveform currently displayed to another waveform filename ("chunk") using the '$W' interactive command.

3) Returns to the data-entry menu, re-enters the FILENAME1 and FRAMESIZE fields, and returns to the display again to show the "chunk" waveform. At this point, 1000 samples (0.125 sec) of data is displayed starting at 0 sec.

4) Returns control to manual mode using the '$MANUAL' interactive command.

See Users Manual section 4.3.5, Writing Sections of Waveforms to a New File, for a fully detailed discussion of the write-to-file command from inside the Waveform Display/Edit function.

### 15.5.3 2-D Spectrograph D/A Output Macro

The following macro example, for real-time software versions only, demonstrates D/A output operation from the 2-D Spectrograph display, and is based on the "playback.mac" macro language file provided on distribution diskette #2. The "playback.mac" macro sends segments of waveform data to the D/A output of the currently installed DSP/Analog hardware (specified by the ANALOG CONVERSION field in the System Configuration menu).

**NOTE: In this example, it is assumed that you have a speaker connected to the first channel specified in the CHANNEL LIST field in the Analog Conversion data-entry menu. The "question" waveform used in the example is located on distribution diskette #3 and contains speech recorded at a 10 kHz sampling rate. This choice is thus suitable for speaker output to easily and conveniently demonstrate the D/A output function, although use of this function is certainly not limited to audio data.**

### 15.5.3 2-D Spectrograph D/A Output Macro

To execute “playback.mac”, first press the ‘F3’ key from the main menu to invoke the Macro Manager menu, and enter ‘playback’ in the MACRO FILENAME field. Then press the <ctrl-Enter> key sequence. Below is the listing of the macro language source file for “playback.mac”:

```
; macro language text file to D/A output from inside 2-D Spectrograph frequency
; domain display function

FFT Generation
   question              ; input filename
   160                   ; analysis framesize = 160 samples
   9                     ; FFT size = 512
   50                    ; 50% overlap
   Hm                    ; Hamming window
   N                     ; no real/imaginary output
   I                     ; use integer precision for speed (this field only
                         ; appears if no DSP Accelerator installed)
   Y                     ; use phase/mag approx. for speed (this field only
                         ; appears if no DSP Accelerator installed)
endfunc

integer length, sampfreq
real total_time, half_time

length = GetHeaderValue(question, len)
sampfreq = GetHeaderValue(question, Fs)
total_time = length / sampfreq
half_time = .5*total_time

2-D Spectrograph
   question              ; input waveform
   blank
   M                     ; magnitude format
   N                     ; no amplitude-squared
   +,N                   ; log-amplitude, linear frequency
   0,0,A                 ; autoscale
   C2                    ; color scheme
   C                     ; contour enabled
   @half_time            ; show half waveform

   $RUN                  ; run

   $D                    ; invoke D/A output

   $T                    ; invoke random-access prompt
   $0.1875 $ENTER        ; enter random-access time value (in seconds)

   $D                    ; invoke D/A output again

   $ESC                  ; return to data-entry menu

   $REPEAT(2) $UP
   $C1                   ; change color scheme to C1
```

```
   $REPEAT(2) $DOWN   ; go down to time shown field
   $0                 ; set time shown field to whole waveform

   $RUN               ; go back into display to see changes

   $D                 ; invoke D/A output again

; use $MANUAL to stay in display continue processing after manual Esc key

   $MANUAL

; remove $MANUAL, and use following sequence to return to main menu, continue
; processing automatically

;  $ESC               ; return to data-entry menu
;  $ESC               ; return to main menu
endfunc
```

There are some interesting notes about execution of the “playback” macro language file, as shown above. First, the macro uses the “general” form of the interactive command to invoke and answer software prompts. The general form includes all text or words not reserved; i.e. not listed as interactive commands in Users Manual section 7.3.2, Issuing Interactive Commands From Macros. In the example above, the ‘$T’ command is used to invoke a random-access time prompt, and then a ‘$0.1875’ command is used to enter a general numeric value. Note that after the numeric entry, a ‘$ENTER’ command is used to indicate the end of the numeric entry and cause the program to complete the prompt and act upon the entry. Second, the macro uses interactive commands to return to the data-entry menu, change a menu field, and re-enter the display. This approach can be useful when changing values specified by the initial data-entry menu field values prior to the start of interactive commands becomes necessary. Third, the macro returns control to manual mode while still inside the display. If the macro contained more processing after the “endfunc” statement, then this processing would resume after you returned to the main menu. If no processing is specified after the “endfunc” statement, then for all intents and purposes, the macro is finished. Note that you can stay inside the display for as long as desired and perform any amount of measurement and other operations, including return to the data-entry menu as many times as necessary; the macro will only resume upon exit to the main menu.

**NOTE: In the macro text above, interactive commands are used to alter and enter some data-entry fields. During normal operation, Hypersignal-Macro initializes various fields in data-entry menus automatically based on information contained in waveform headers. This usually happens when a data-entry menu is first entered, requiring the program to "load" a waveform specified in a waveform filename field. It can also happen when you change waveform filename field entries inside a data-entry menu. This type of operation poses a basic conflict with macro operation, since often times data-entry menu fields are specified in the macro. When a function or display is invoked from a macro, all data-entry fields not marked 'no_change' or 'don't_care' take precedence over the automatic load and initialize. *However*, if, from inside the macro file, you use interactive commands to cause another waveform load (by causing a filename entry field to be "entered"), then the automatic load will take precedence, and one or more data-entry fields containing previous macro file inputs may be overwritten. Using interactive commands to enter waveform filenames must be done carefully and is not usually recommended.**

The "playback" macro source file performs the following major steps:

1) First performs an FFT Generation operation on the "question" speech waveform (provided on distribution diskette #3) to ensure a frequency domain waveform to serve as a convenient example.

2) Sends the segment of the waveform currently displayed to the D/A output of the current DSP/Analog hardware. Since half of the waveform was initially displayed (see calculations for variable "half_time" entered in the TIME SHOWN subfield), the first half of the waveform will be heard over the speaker.

3) Repositions the start of the 2-D Spectrograph display for "question" to 0.1875 sec using the random-access time prompt ('$T' interactive command). The combination of TIME SHOWN data-entry menu field entry and the random-access time prompt can be used to exactly position a desired segment of waveform inside the current display.

4) Again sends the segment of the waveform currently displayed to the D/A output of the current DSP/Analog hardware.

5) Returns to the data-entry menu, changes the COLOR field to a different color scheme (why not?), sets the TIME SHOWN subfield to ‘0’ to specify display of the entire waveform, and returns to 2-D Spectrograph display.

6) Agains sends the segment of the waveform currently displayed to the D/A output of the current DSP/Analog hardware.

7) Returns control to manual mode using the ‘$MANUAL’ interactive command.

See Users Manual section 4.3.3, D/A Playback From Waveform Displays, for a fully detailed discussion of the D/A output command from inside either the 2-D Spectrograph display or Waveform Display/Edit functions.

## 15.5.4 Instrumentation Display Macros

The following short macro examples demonstrate how to invoke the Digital Oscilloscope and Spectrum Analyzer instrument displays from a macro language file. *These examples are valid for real-time versions of software only.* The macro listings shown below require no input parameters. The first example demonstrates waveform file input:

```
; example macro language text files to show how to use instrumentation displays
; (Spectrum Analyzer and Digital Oscilloscope) from macro language

SPECTRUM ANALYZER
   speech                ; waveform file input on trace 1
   blank                 ; no input on trace 2
   m+                    ; trace 1 is in log magnitude format
   don't_care            ; don't care about trace 2 format
   10^3,20               ; frequency axis is log10, with 20 Hz minimum cutoff
   50000                 ; 50 kHz sampling frequency
   5                     ; ⌠5 full-scale voltage range
   10,512,40             ; FFT length = 2^10 = 1024, framesize = 512,
                         ; 3D frames = 40
   Hm                    ; Hamming Window
   2d,d,lin,i            ; 2D format, dual-trace (ignored), linear draw,
                         ; intersected grid

   $RUN                  ; run
   $REPEAT(20) $UP       ; move crosshair cursor up
   $REPEAT(50) $LEFT     ; move cursor to right
   $ESC                  ; return to data-entry menu
   $ESC                  ; return to main menu
endfunc
```

The second example demonstrates waveform analog input, using the Digital Oscillscope function:

```
DIGITAL OSCILLOSCOPE
   a0                ; channel 0 input on trace 1
   blank             ; no input on trace 2
   50000             ; 50 kHz sampling frequency
   1,0               ; initial trigger trace is 1, initial level is zero
   5                 ; ⌠5 full-scale voltage range
   o,lin,i           ; dual-trace display (ignored), linear draw, intersected grid
   y                 ; axes enabled

   $RUN              ; run
   $REPEAT(3)        ; scale waveform up
   $PGUP
   $REPEAT(100)      ; let 100 frames display
   $DISPFRAME
   $REPEAT(3)        ; increase timebase
   $LEFT
   $REPEAT(100)      ; let 100 frames display
   $DISPFRAME
   $REPEAT(6)        ; decrease timebase
   $RIGHT
   $REPEAT(100)      ; let 100 frames display
   $DISPFRAME
   $ESC              ; return to data-entry menu
   $ESC              ; return to main menu
endfunc
```

The significant aspect of the example above is the interactive '$DISPFRAME' command. Each time this command is executed, another analog frame is allowed to display, or "pass through the gate", by the program. Prefixing this command with the $REPEAT command, which can accept a 32-bit integer argument, enables control over the number of analog frames that are displayed. For unlimited display, the $MANUAL interactive command can be used, as in the example below:

```
DIGITAL OSCILLOSCOPE
   speech            ; waveform file input on trace 1
   a1                ; channel 1 input on trace 2
   50000             ; 50 kHz sampling frequency
   1,0               ; initial trigger trace is 1, initial level is zero
   5                 ; ⌠5 full-scale voltage range
   o,lin,i           ; dual-trace display (ignored), linear draw, intersected grid
   y                 ; axes enabled

   $RUN              ; run
   $MANUAL           ; yield control inside instrument display to user
```

```
endfunc

...continued macro processing...
```

In the above example, if there was subsequent macro processing after the ‘endfunc’ statement for the Digital Oscilloscope, such processing would resume *after* the user pushed the ‘Esc’ key from inside the display, and then again from the data-entry menu.

## 15.6 Invoking External Programs From a Macro

This section contains some text examples of how to invoke external programs from inside a macro. In the first example, it is assumed that the UTIL FILE,MENU LABEL1 field in the External Program Configuration menu selection is set up as follows:

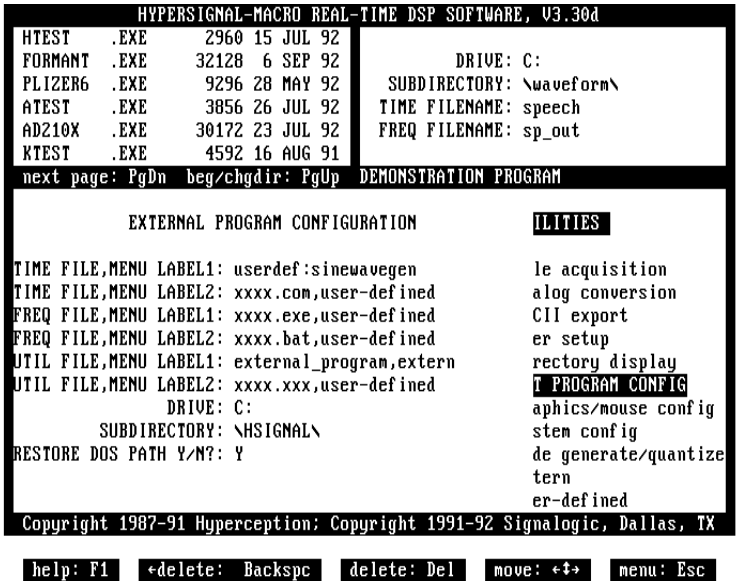


*Figure 15-5*

In the menu above, the main menu label has been specified as “external_program” and the filename of the external program as “extern” (the program searches for .COM, .EXE, and .BAT filename extensions, in that order). Below are some examples of macro language text that should

serve as a guide to writing a macro that would invoke an external program. If you have Hypersignal-Acoustic software, an excellent, actual example is included on distribution diskette #2 in the form of the macro language source file “TEFRD.MAC”.

```
; Example of macro language text that would invoke external program “extern”
; and pass command-line parameters

integer A

External_Program
   “filename”              ; pass a string
   @A                      ; pass the value of a variable as a string
   #1                      ; pass an input parameter as a string
endfunc

User-Defined Util 1   ; does same thing, but with generic name
   “filename”
   @A
   #1
endfunc
```

In the above examples, the parameters between the function name and the “endfunc” statement, which are normally menu field parameters, are treated by the macro language compiler as command-line parameters to be sent to the external program when it is executed. Up to 10 command-line parameters may be passed in this manner. Note that all macro language variable, constants, input parameter references, and string function references can be used, and will be evaluated at run-time, just prior to being passed as command-line parameters.

Note that the second example above would be appropriate in a case where the External Program Configuration menu had to be set up *during* the macro, and the name of the external program would thus not be known at compile-time.

*this page intentionally left blank*

# 16.0 LPC AUTOCORRELATION

This chapter of the tutorial briefly presents the why and how of LPC autocorrelation. LPC autocorrelation is often used on periodic or semiperiodic data to construct a series of filters which have frequency responses that represent the major frequency domain energy areas in the input data. The input data is segmented into frames, and a filter is constructed on each frame of input data, to allow for time-varying characteristics in the input data. The resulting series of “analysis filter” coefficients can be used to create a series of impulse responses, which, when strung together, represnt a “facsimile” of the input data. A form of compression is inherent in this process, because only the filter coefficients are needed to represent the original input data, with the overall compression ratio being the iinput framesize divided by the number of coefficients. And, the resulting series of frequency response frames, resulting from either the filter coefficients directly or the synthesized data is “smooth”, showing only the major energy areas.

First we will apply LPC autocorrelation to the “speech” waveform included on distribution diskette #3, using “analysis filter” coefficient selection. Then we will use the FFT Generation function to calculate frequency responses for both the original data and the newly created LPC autocorrelated data. Finally, we will compare the two frequency responses in the 2-D Spectrograph and 3-D Spectrogram frequency domain displays. To begin, enter the following LPC Autocorrelation menu:

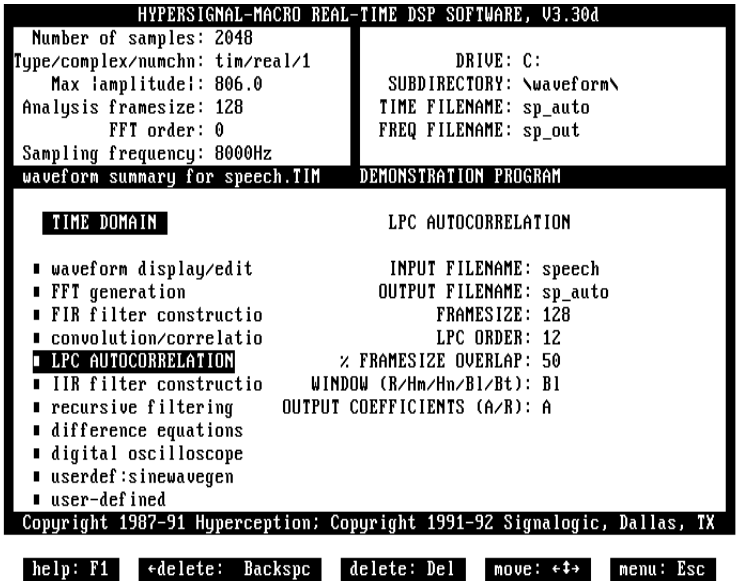


*Figure 16-1*

Remember to enter the last field or press <ctrl-Enter> to cause processing to begin. In the above example, we are using a FRAMESIZE of 128, a filter size of 12 coefficients (each output frame will consist of 12 samples), an overlap of 50%, a Blackman window, and have selected “analysis filter” coefficients (‘A’ entry in OUTPUT COEFFICIENTS field).

The choice of framesize depends completely on your data and your application. Generally, FRAMESIZE should be chosen such that you feel your data remains “quasi-stationary” for this many samples (a duration of time equal to FRAMESIZE divided by the sampling frequency of the waveform). By quasi-stationary, we mean that the major frequency components in the data during that period of time didn’t change suddenly; by segmenting the input waveform into successive segments of FRAMESIZE length, slowly changing frequency characteristics can be analyzed with respect to time. The ‘Bl’ window entry will eliminate any “edge” noise due to the segmenting of the data into arbitrary frames (‘Bl’ invokes a Blackman window, which tapers smoothly to zero at both ends of the FRAMESIZE length). The choice of 50% FRAMESIZE OVERLAP ensures that no noise spikes, glitches, or other small (down to single-sample duration) data features will be missed becaused they occurred at or near the frame edges, where the window value is zero or nearly zero.

See Users Manual section 4.4.1, Framesize and Overlap Discussion, for more information on this subject.

Allow the above LPC Autocorrelation menu to complete, then enter the following FFT Generation menu:

| | |
|---|---|
| FILENAME: | sp_auto |
| FRAMESIZE: | 13 |
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 0 |
| WINDOW (R/Hm/Hn/Bl/Bt): | R |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields only apply if you are using a non-real-time software version, or you have the DSP ACCELERATOR field disabled ('none' entry) in the System Configuration menu.

**NOTE: In the above FFT Generation menu, the framesize is the LPC Autocorrelation filter length + 1, overlap is zero, and no window has been specified (i.e. a rectangular window). This is because the framesize, overlap and window analysis has already been done in the LPC Autocorrelation function, and the results must not be "taxed twice". Also, note below that when we perform FFT Generation on the unfiltered data, we use *identical* framesize, overlap and window parameters as in the LPC Autocorrelation function.**

After the FFT Generation menu above finishes, return to the Time Domain column and enter another FFT Generation menu:

| | |
|---|---|
| FILENAME: | speech |
| FRAMESIZE: | 128 |
| FFT ORDER: | 9 |
| % FRAMESIZE OVERLAP: | 50 |
| WINDOW (R/Hm/Hn/Bl/Bt): | Bl |
| REAL/IMAG OUTPUT (Y/N): | N |
| PRECISION (I/R): | R |
| PHASE/MAG APPROX (Y/N): | N |

Remember that the PRECISION and PHASE/MAG APPROX fields only apply if you are using a non-real-time software version, or you have the

DSP ACCELERATOR field disabled (‘none’ entry) in the System Configuration menu.

Now we are ready to comparison view the filtered and unfiltered data in the frequency domain. Enter the following 3-D Spectrogram display menu:

```
                    FILENAME1: sp_auto
                    FILENAME2: speech
              POWER SPEC (P/M): M
             AMP SQUARE (Y/N): N
         LOG (N/+/-,N/base,Min): -
       SURFC, GRID(H/M/N,N/S/I): H,N
       Y-LABELS (SCALE,UNITS): 1,Units
  AMP SCALING (TOP,BOT,REF): 0,0,A
  ANGLE,TIME SHOWN (5-90, S): 45,0
```

Note that the software automatically initializes the LOG field above to negative log display. Negative log display mode is characteristically required when displaying the frequency response of LPC autocorrelation filter coefficients directly (as opposed to displaying the frequency response of data synthesized using the coefficients). Note also that the software will *not* invert the “speech” waveform prior to display, and will use positive log display instead, because the assumption is that you are trying to compare LPC autocorrelated data with un-autocorrelated data. (If both waveforms represented LPC autocorrelation filter coefficients, then the software would use negative log display for both). After ensuring the above menu fields are entered, and pressing the <ctrl-Enter> key sequence, you should see a display similar to Figure 16-2, below:

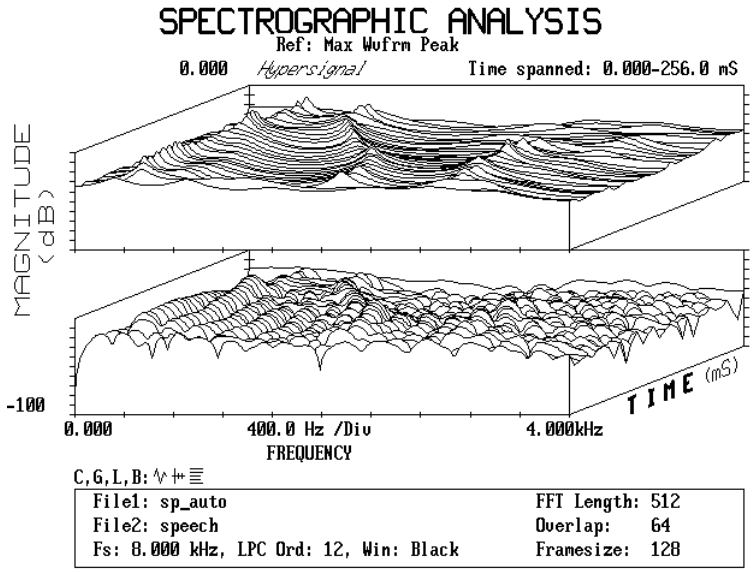


*Figure 16-2*

Now, exit to the main menu and enter the following 2-D Spectrograph display menu:

```
                    FILENAME1: sp_auto
                    FILENAME2: speech
                POWER SPEC (P/M): M
                AMP SQUARE (Y/N): N
             LOG (N/+/-,N/base,Min): -
AMP SCALING (TOP,BOT,REF): 0,0,A
                COLOR (Cn/Gn/M): C2
                  SURFACE (C/R): C
       TIME SHOWN (S, 0=ALL): 0
```

Again, the LOG field is initialized as explained above. You should see a display similar to Figure 16-3, below:

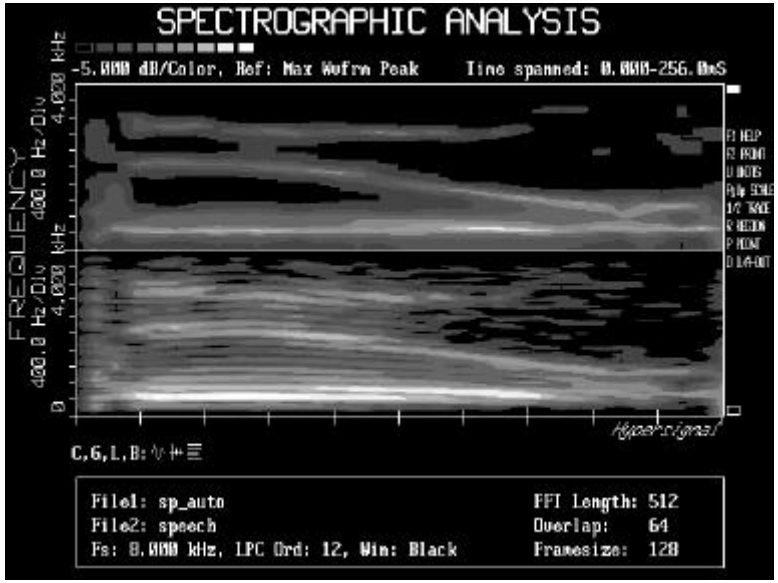


*Figure 16-3*

Can you see the major energy areas that have been extracted in the top trace? Note the smoothness and clarity of the LPC autocorrelated trace, which indicates its value in various processing algorithms, especially for frequency domain feature extraction purposes. In this case, the input data was speech (the spoken word “two”), so there are three or four major “formants”, which move up and down in frequency as time changes. The change in frequencies represents the changes in in the speaker’s vocal tract resonances as the word “two” was uttered.

*this page intentionally left blank*